Debreceni Egyetem
Informatikai Kar

Felületábrázolás és alkalmazásai Maple-ben

Témavezető:
Dr. Hoffmann Miklós
egyetemi docens

Készítette:
Szlahorek András
informatikatanár

Debrecen
2009

Tartalomjegyzék

1. Előszó

Napjainkban egyre több olyan problémával találkozunk a tudományok számos területén, amelyet hagyományos eszközökkel (papír, ceruza) nem, vagy csak igen nehezen tudnánk megoldani. A számítógép elterjedésével azonban ezen problémák nagy részének a megoldása lehetővé vált. Például mérnöki és műszaki tudományok. Az utóbbi évtizedekben az informatika rohamos fejlődésével kezdtek megjelenni az úgynevezett *számítógépes algebrai rendszerek*. A *számítógépes algebrai rendszerek* olyan komplex rendszerek, amelyek számos tudományágban dolgozó embernek segítséget nyújtanak a matematikai számítások elvégzéséhez.

Manapság egyre fontosabbá válik a tudományok és tantárgyak közötti kapcsolat és átjárhatóság. Gondoljunk arra, hogy ma egy matematika tanárnak (természetesen nem csak matematika tanárról beszélhetünk) nem elég csak a matematika területén felkészültnek lenni. A matematika oktatásban is használt tudjuk venni a számítógépnek, sőt egyes esetekben hasznosabb bármilyen más eszköznél. Elég ha csak megemlítjük a függvények vizsgálatát és megjelenítését, felületek ábrázolását, különböző geometriai modellezéseket, de nem szabad elfelejteni a differenciál és integrálszámítást, a lineáris algebra témaköreit és még sorolhatnánk. Az egyes szabályok és fogalmak kialakításában is sok segítséget nyújt a számítógép (pl. függvény szélsőérték helyei és a derivált függvény kapcsolata). Persze nem elég önmagában a számítógép, megfelelő programokra is szükség van. Ebben segít nekünk a Maple.

A Maple egy olyan számítógépes algebrai rendszer, amelynek ha nem is az általános iskolások (bár némely esetben ők is), de a középiskolásoktól kezdve minden korosztály használt tudja venni a tanulásban vagy a munkában. A Maple felhasználóbarát, szinte bárki el tudja sajátítani a kezelését.

A szakdolgozat célja elsődlegesen a Maple grafikai lehetőségeinek bemutatása. A szakdolgozat első részében arról lesz szó, hogy a Maple milyen eszközöket kínál nekünk a grafikus megjelenítéshez, szó lesz a 2 dimenziós grafikáról is, bár a szakdolgozat inkább a 3 dimenziós grafika köré épül. Említésre kerülnek a Maple grafikai csomagjának számos rendkívül jól használható függvényei, példákat mutatok az alkalmazásukhoz, illetve az ábrák saját igényeinknek megfelelő átalakításához. Ezen használati útmutató jellegű rész után a másodrendű felületek megjelenítésével foglalkozom. Kitérek részletesen a másodrendű felületek síkmetszeteire, azokat szemléletes animációkkal igyekszem bemutatni, végül a másodrendű felületek áthatásairól lesz szó.

A szakdolgozat olvasásához és megértéséhez szükséges némi matematika ismeret (leginkább az ábrázoló geometria területéről), illetve a Maple rendszer alapvető használatának ismerete (pl. függvények definiálása, egyenletek megoldása, az alapvető adatstruktúrák ismerete, különböző polinom átalakítási parancsok használata, stb.).

A szakdolgozat hasznos lehet azoknak a diákoknak illetve tanároknak, akik szeretnének geometria objektumokat a számítógép segítségével megjeleníteni, görbéket vagy felületeket ábrázolni, illetve ezekkel kapcsolatosan szemléltető animációkat készíteni, például egy geometria óra megtartásához. Mindazonáltal örömmel forgathatják azok is, akik egyszerűen meg szeretnék ajándékozni magukat egy kis esztétikai örömmel, a szakdolgozat ábráit szemlélve.

2. Bevezetés a Maple grafikájába

Mielőtt hozzákezdenénk függvényeink ábrázolásához, nem árt tudni, hogy milyen lehetőségeket ad a kezünkbe a Maple, mit lehet ábrázolni, illetve az ábrázolást a háttérben hogyan valósítja meg a Maple.

2.1. Grafikai lehetőségek Maple-ben

A Maple számos olyan parancsot ad nekünk, amivel lehetőségünk van grafikus ábrázolásra. A Maple grafikus függvényeket tartalmazó csomagja a `plots`. Ezeknek a függvényeknek a használatához szükséges a `plots` csomag betöltése a `with(plots):`-val. Leggyakrabban a `plot` és a `plot3d` függvényeket fogjuk használni, ehhez azonban nem szükséges az előbbi csomag betöltése. A `plots` csomag függvényeinek a használatához sem szükséges a teljes csomag betöltése, hivatkozhatunk a csomag egy függvényére a következő formában is:

```
>plots[függvénynév](argumentumok);
```

A `plots` rengeteg jól használható függvényt tartalmaz, amelyekkel lehetőségünk van például differenciálegyenletek megoldásainak, vektormezőknek, komplex függvényeknek, statisztikai számítások eredményeinek, stb. megjelenítésére.
Nézzük meg, hogy a Maple segítségével 2D-ben és 3D-ben miket ábrázolhatunk:

2D:
- egyváltozós valós függvényekkel definiált görbék
- paraméteres egyenletekkel definiált görbék
- egyenlettel (implicit módon) megadott görbék
- adathalmazokhoz tartozó szintvonalas és sűrűségi ábrák
- vektormezők
- statisztikai adatok
- poligonok, körök és egyéb geometriai alakzatok
- 2D-s objektumok animációja

3D:
- kétváltozós valós függvénnyel definiált görbék
- paraméteres alakban megadott felületek és görbék
- egyenlettel (implicit módon) megadott felületek
- poliéderek, gömbök, tóruszok és egyéb geometriai objektumok
- 3D-s objektumok animációja

Mielőtt rátérnénk a tényleges ábrázolásra, fontos megemlíteni egy pár szóban, hogy a Maple hogyan dolgozik a háttérben, milyen grafikai struktúrát használ a megjelenítéshez.

2.2. A Maple grafikai struktúrája

A Maple-ben az ábrák elkészítése két lépésben történik. Tegyük fel, hogy egy grafikus függvényt meghívtunk. Az első lépés, hogy a Maple létrehoz egy adatstruktúrát, ami tartalmazza az összes szükséges információt (az ábrázolandó függvényt, a kiszámított pontokat, a tengely stílusát, színeket, stb.) a grafikus objektum megjelenítéséhez. Kétdimenziós ábráknál ezen adatstruktúra neve `PLOT`, háromdimenziós ábráknál `PLOT3D`. A második lépésben a grafikus objektum megjelenik a képernyőn. A `PLOT` és a `PLOT3D` adatstruktúrákat úgy tekinthetjük, mint bármely más Maple kifejezést.
Ezek az adatstruktúrák a következő formát veszik fel:

függvénynév(grafikus objektum(ok), globális paraméterek)

A grafikus objektumok alakja pedig:

grafikus objektum(interpolációs adatok, objektum paraméterek)

Ezen struktúrákhoz tartozó ábrákat a `display` parancs segítségével tudjuk megjeleníteni.
Egy ilyen struktúrát úgy írathatunk ki, ha valamilyen grafikus parancs eredményét egy változónak értékül adjuk és a végére `;`-t írunk.

```
>p:=plot([[2,3],[4,5]],color=red);
```

$$p := INTERFACE\_PLOT(CURVES([[2.,3.],[4.,5.]]), AXESLABELS("",""),$$
$$COLOUR(RGB, 1.00000000, 0., 0.), VIEW(DEFAULT, DEFAULT))$$

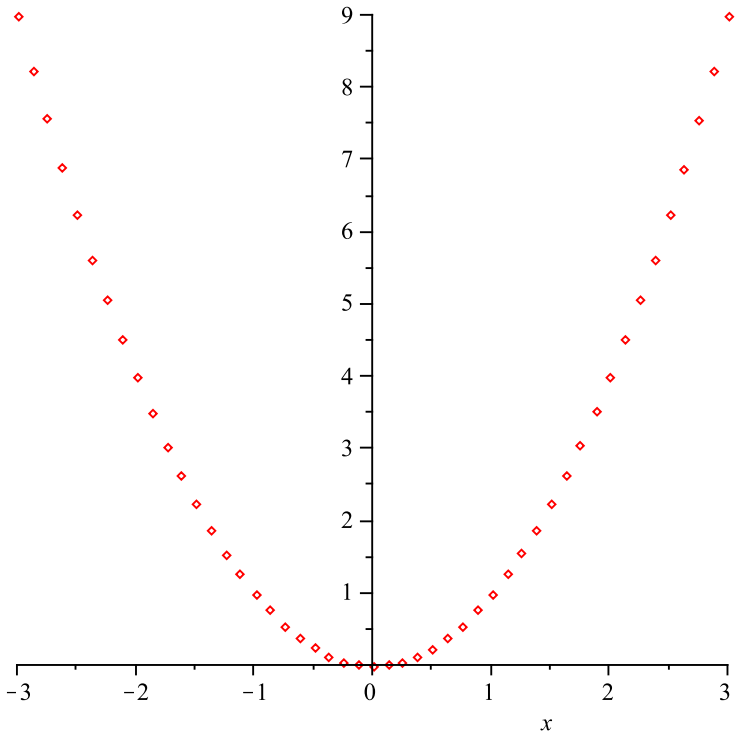
Láthatjuk, hogy a struktúránk a `plot` interfészt használja, azon belül van egy CURVES objektum, azt követik a paraméterek és értékeik.
Amennyiben például egy $f(x) = x^2$ függvényhez tartozó struktúrát íratnánk ki, rengeteg pontpárt (x és y értékkel) láthatnánk, szám szerint 49-et (a 49 pontot szemléltető grafikon: 1. ábra):
A Maple úgy tud szép sima függvényeket kirajzolni, hogy egy bizonyos algoritmus szerint növeli a mintapontok számát, amennyiben az szükséges. Kezdetben (az alapértelmezett) 49 mintapont van. Ezeket a pontokat egyenes szakaszokkal köti össze és ha 2 szomszédos szakasz találkozásánál túl nagy a törés (nagy a 2 szakasz által bezárt szög), akkor növeli a mintapontok számát. Ezt nevezik adaptív ábrázolásnak. A mintapontok számának növelése nem mehet a végtelenségig, a határ 200 mintapont.
A `display` parancs segítségével már egy létező grafikus struktúra megjelenését is befolyásolhatjuk. A következő példában a szakaszunk megjelenítését korlátozzuk a megadott x és y tartományra (2. ábra):

```
>display(p,view=[-3..3,0..9]);
```

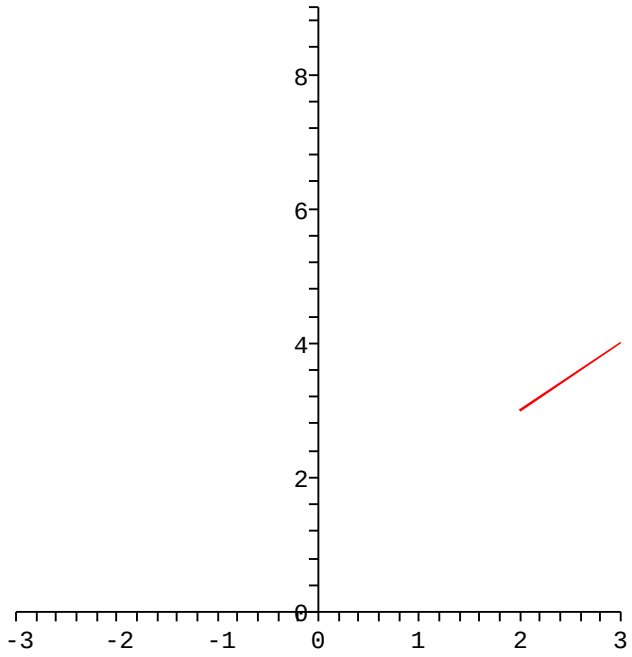
Vannak olyan jellemzők, amit már nem lehet utólag a `display` parancs segítségével módosítani, de egy kis trükkel a módosításra még így is lehetőség van. Az x^2 grafikonunk alapértelmezés szerinti színe piros, ezt utólag a `subs` parancs segítségével változtathatjuk meg a megjelenítés idejére, mivel a `subs` csak helyettesít, értéket nem változtat (3. ábra):

```
>display(subs(COLOUR(RGB,1.00000000,0.,0.)=COLOUR(RGB,0,0,1),p));
```

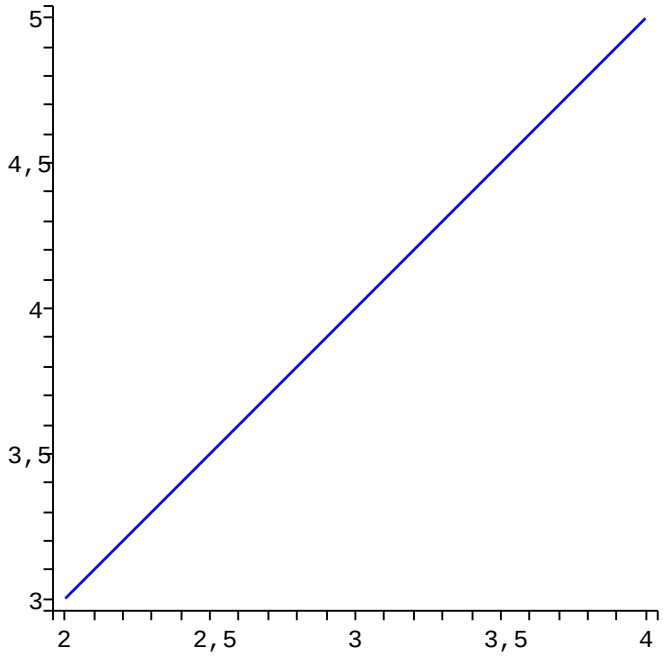
A `display` parancs akkor is hasznos, ha több már korábban elkészült ábrát vagy animációt szeretnénk közös koordináta-rendszerben megjeleníteni.



1. ábra. Az x^2 grafikonja a 49 ponttal



2. ábra.



3. ábra.

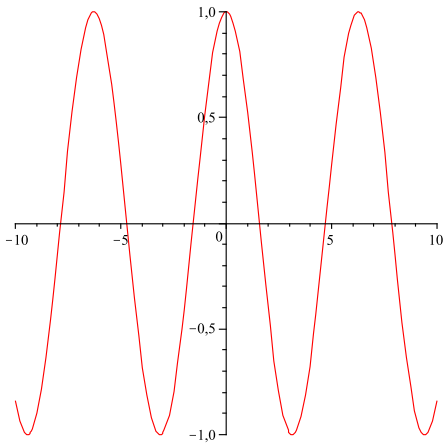
3. A plot és a plot3d parancsok Maple-ben

A Maple által kínált számos grafikus parancs közül a leggyakrabban használt 2 parancs a `plot` és a `plot3d`.
Ebben a részben főleg ennek a 2 parancsnak a használatát és a hozzátartozó paramétereket ismerhetjük meg, melyekkel az általunk kívánt formában jeleníthetjük meg függvényeinket. Értjük ez alatt az ábrák színét, feliratok beállítását, skálázást, árnyékolást, projekciót, világítást és így tovább.

3.1. A plot parancs

Egyváltozós függvényeket a plot parancs segítségével jeleníthetünk meg. Legegyszerűbb megadása : plot(függvénynév);

```
>plot(cos); (4. ábra)
```



4. ábra.

```
>plot(x->cos(x)):
```

A másik lehetőség, hogy kifejezést adunk, ekkor már az értelmezési tartományt is meg kell adni.

```
>plot(cos(x),x=-2*Pi..2*Pi):
```

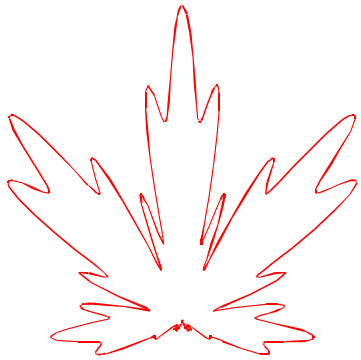
Az értelmezési tartománynál megadhatjuk a végtelent jelentő `infinity` szót is.
Egyszerre több ábrázolandó függvényt is megadhatunk, amit listaként vagy halmazként kell átadni a `plot` számára. A függvényt paraméteres alakban is megadhatjuk:

```
>plot([f(t),g(t),t=a..b],opciók);
```

Egy érdekes függvény a Maple juharlevél szimbólumát rajzolja ki(5. ábra):

```
>s:=t->100/(100+(t-Pi/2)^8):

>r:=t->s(t)*(2-sin(7*t)-cos(30*t)/2):
```

```
>plot([r,t->t,-Pi/2..3/2*Pi],coords=polar,axes=none);
```



5. ábra.

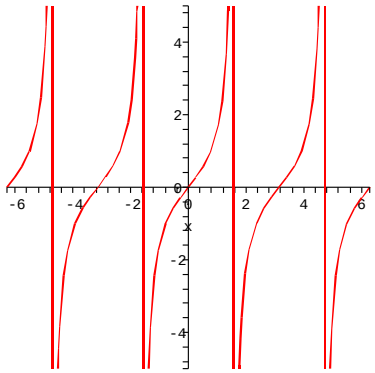
3.2. A plot opciói

Skálázás (scaling): A koszinusz függvény grafikonjánál láthattuk, hogy az y tengelyen 2 egység, az x tengelyen 4*Pi egység van, de a hosszuk megegyezik a megjelenítésnél. Hogy méretarányos legyen a grafikon, azt a `scaling=constrained` opció megadásával tehetjük meg. Az alapértelmezés `unconstrained`.

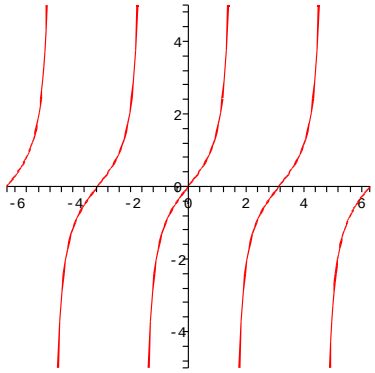
Megjelenítés korlátozása (view): Némely esetben szükség lehet a függőleges tartomány korlátozására is. Ezt egyszerűen megtehetjük a függőleges tartomány megadásával vagy a `view` paraméterrel (6. ábra):
`>plot(tan(x),x=-2*Pi..2*Pi,-5..5):`
vagy
`>plot(tan(x),x=-2*Pi..2*Pi,view=-5..5);`
A `plot3d` parancsnál a `view` paramétert így is megadhatjuk:
`view=[xmin..xmax, ymin..ymax, zmin..zmax]`.

Szakadások eltüntetése (discont) és koordináta-rendszer típusa(coords): A Maple a tangens függvény képénél berajzolta nekünk az aszimptotákat is. A `discont=true` opcióval a Maple kiküszöböli a szakadásokból adódó pontatlanságokat (7. ábra):
`>plot(tan,-2*Pi..2*Pi,-5..5,discont=true);`
Amennyiben az alapértelmezett Descartes-féle koordináta rendszertől eltérőt akarunk használni, azt a `coords=típus` paraméterrel adhatjuk meg. Használhatunk: `bipolar`, `cassinian`, `elliptic`, `hyperbolic`, stb. rendszereket (8. ábra):
`>plot([sin,cos,-sin,-cos],-Pi..Pi,coords=polar);`

Tengelyek beosztásai (xtickmarks, ytickmarks) és címek (title): A tengelyeken az egyes beosztásoknál a számunkra megfelelő feliratok elhelyezéséhez használjuk az `xtickmarks` és `ytickamrks` opciókat. Egy listát kell megadnunk,



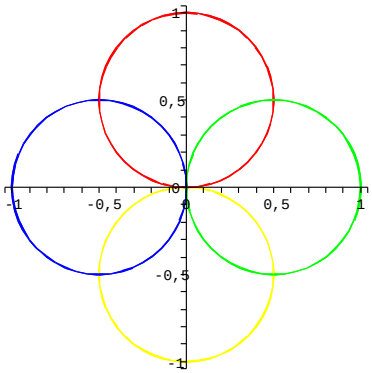
6. ábra.



7. ábra.

amelyben érték és sztring párok szerepelnek, amik meghatározzák, hogy a tengelyen az adott értéknél milyen felirat legyen. Például:
`>plot(tan(x),x=-Pi..Pi,-5..5,discont=true,xtickmarks=[-3.14="-Pi", -1.57="-Pi/2",1.57="Pi/2",3.14="Pi"], ytickmarks=3);`
Egész számot megadva a tengelyen lévő osztáspontok számát állíthatjuk be.
Az ábránk címét a `title=cím` opcióval adhatjuk meg, a betűtípust, stílust és méretet is beállíthatjuk a `titlefont=[betűtípus,stílus,méret]` lista megadásával.

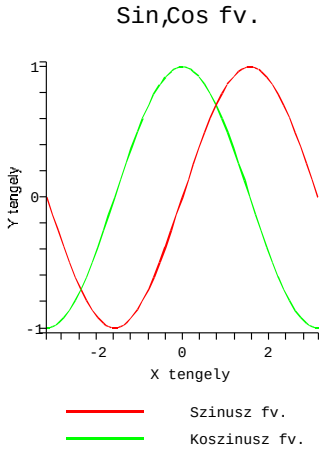
Tengelyek típusai (axes), feliratai (labels) és beállításai: Megadhatjuk a tengelyek típusát is az `axes=típus` paraméterrel. A típus lehet `FRAME`, `BOXED` (vagy `box`), `NORMAL` és `NONE`.
Az `axesfont=[beállítások]` segítségével a tengelyek betűtípusát, stílusát, méretét állíthatjuk be.
A `labels=[x,y]` segítségével az x és y tengely feliratait állíthatjuk be, a `labeldirections=[xirany,yirany]` -val az x és y tengely feliratának irányát (`HORIZONTAL, VERTICAL`), a `labelfont=[beállítások]` segítségével a tengely feliratainak típusát, stílusát és méretét állíthatjuk be.



8. ábra.

A `legend=sztring` segítségével a grafikon feliratát adhatjuk meg, több függvény esetén a feliratokat listában kell megadni. Nézzünk minderre egy példát (9. ábra)

```
>plot([sin(x),cos(x)],x=-Pi..Pi,legend=["Szinusz fv.",
"Koszinusz fv."], title="Sin és Cos fv. képei", titlefont=[ARIAL,
BOLD,16], labels=["X tengely","Y tengely"], labeldirections=
[horizontal, vertical],axes=frame, xtickmarks=4, ytickmarks=3);
```

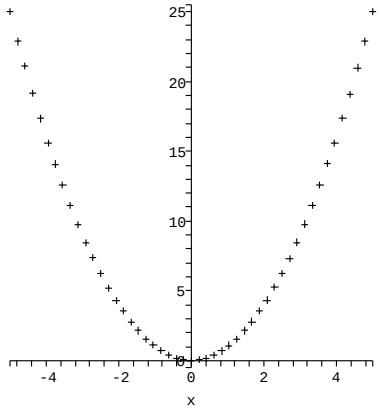


9. ábra.

Mintapontok számának növelése (numpoints): A 2D-s grafikonok megjelenítésekor a Maple adaptív ábrázolást (lásd. 2.2) alkalmaz. Alapértelmezésben a függvény kirajzolásához 50 mintapontot készít, de ezt mi is beállíthatjuk a `numpoints=n` paraméterrel. Az `adaptive=false` opcióval az adaptív ábrázolást kikapcsolhatjuk.

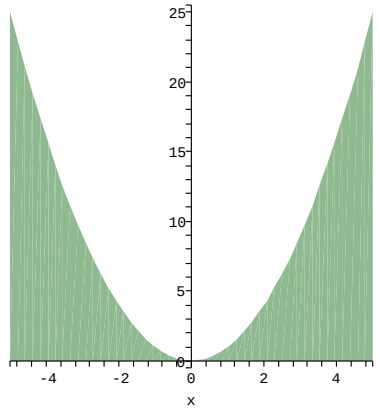
Görbestílus (style), vastagság (thickness), szín (color) és kitöltés (filled): Lehetőség van a függvénygörbe stílusának megváltoztatására a `style=stílus` paraméterrel, ahol a `stílus` lehet: `LINE`, `POINT`, `PATCH`, `PATCHNOGRID`. A két

utóbbi lehetőséget poligonok ábrázolásánál használhatjuk.
A függvénygörbe pontjait szimbólumokkal is megjeleníthetjük a `symbol=s` -sel, ahol `s` lehet `BOX`, `CROSS`, `CIRCLE`, `POINT`, `DIAMOND`. A `symbolsize=n` -nel a méretét is változtathatjuk (10. ábra).
`>plot(x^2,x=-5..5,style=point,symbol=cross);`
A kirajzolt függvénygörbe vastagságát a `thickness=n` paraméterrel állíthatjuk,



10. ábra.

ahol `n` egy nemnegatív egész érték, a vonal stílusát a `linestyle=stílus` -sal módosíthatjuk, ahol a stílus lehet: `SOLID`, `DOT`, `DASH`, `DASHDOT`.
`>plot(x^3,x=-5..5,thickness=3,linestyle=DOT,numpoints=10):`
A függvénygörbe színét a `color=c` paraméterrel állíthatjuk. Vannak előre definiált színek (összesen 21), illetve mi is megadhatunk saját színeket RGB vagy HUE színskálát használva. Lehetőség van általunk választott szín saját névvel történő ellátására is a `macro` függvény segítségével. Lehetőség van a függvénygörbe és az x tengely közötti terület kitöltésére is a `filled=true` paraméter megadásával. Nézzünk egy példát az előbbiekre (11. ábra):
`>plot(x^2,x=-5..5,color=COLOR(RGB,0.56,0.73,0.56),thickness=3);`



11. ábra.

3.3. Egyéb 2D-s függvények

Vektorok ábrázolása: Kétdimenziós és háromdimenziós vektorok ábrázolására az `arrow` függvényt használhatjuk.
`>arrow(u,opciók);`

`u`: vektorok halmaza vagy listája

`opciók`: beállíthatjuk a vektorok alakját (`shape=harpoon`, `arrow`, `double_arrow`, `cylindrical_arrow`), a hosszát (`length`), szélességét (`width`), a vektor hegyének szélességét (`head_width`) és hosszát (`head_length`).

```
>arrow([<1,4>,<5,6>]):
>arrow([seq(<i,i-5,i+7>,i=1..5)]):
```

2D-s vektormezőt is ábrázolhatunk, erre szolgál a `fieldplot` és a `gradplot`.
`>fieldplot(t,v,f);`

`t`: vektor(ok)

`v`: vízszintes tartomány

`f`: függőleges tartomány

```
>fieldplot(<sin(x),cos(y)>,x=0..8,y=0..8):
```
A `gradplot` is hasonlóan működik, csak ott vektor helyett függvény(ek) szerepel.
```
>gradplot(x^2+y^2,x=-4..4,y=-4..4):
```

Komplex függvények ábrázolása: Komplex függvényeket a `complexplot` paranccsal ábrázolhatunk.
`>complexplot(f,p,v);`

`f`: a komplex függvény

`p`: a komplex függvényben levő paraméter tartománya

`v`: az ábrázolás vízszintes tartománya

```
>complexplot(sin(x+I),x=-Pi..Pi):
```

Sűrűségábrázolás: A `densityplot` segítségével egy kétváltozós függvény értékeit árnyalatokkal tudjuk ábrázolni. Hasonló a `contourplot` függvényhez, de míg a `contourplot` ugyanezt vonalakkal rajzolja ki, addig ez a parancs ezt szürkeárnyalatos képpel szemlélteti.
```
>densityplot(kifejezés, x=a..b, y=c..d);
```
A kifejezésben szerepelnie kell az x és y változóknak.
```
>densityplot(x^2+y^2,x=-5..5,y=-5..5):
```

```
>densityplot(sin(x)+cos(y),x=-5..5,y=-5..5):
```
Az alapértelmezés a fekete-fehér árnyékolás (`SHADING`), de ezt meg is változtathatjuk a `colorstyle` paraméterrel. Lehet: `HUE`, `RGB` vagy tetszőleges szín árnyalatai.

```
>densityplot(sin(x)+cos(y),x=-5..5,y=-5..5,color=red):
```

Implicit görbék ábrázolása:
`>implicitplot(kifejezés,x=a..b,y=c..d);`
A `kifejezés` x-et és y-t tartalmazza. Nézzünk egy példát hiperbolára:
`>implicitplot((x-3)^2/2-y^2/2=1,x=-3..9,y=-10..10):`
Ellipszis:
`>implicitplot(x^2/9+y^2=4,x=-9..9,y=-4..4):`

```
>implicitplot(x^3+y^3-8*x*y=-1,x=-5..5,y=-5..5):
```

Logaritmikus grafikonok: Logaritmikus grafikonok készítésére a `logplot` és a `loglogplot` függvények szolgálnak. A `logplot`-nál a függőleges tengely logaritmikus skálázású, a `loglogplot`-nál mindkét tengely.
`>logplot(sin(x),=0..2*Pi):`

```
>loglogplot(sin(x),x=0.1..2*Pi):
```

Poligonok ábrázolása: Poligonok ábrázolását a `polygonplot` paranccsal tehetjük meg, melynek paraméterben kell megadni a poligon csúcsait.
`>polygonplot(cs,opciók);`

`cs`: lista vagy halmaz a poligon csúcspontjaival
`>polygonplot([[10,0],[5,3.3],[5,10],[0,6.6],[-5,10],[-5,3.3],`
`[-10,0],[-5,-3.3],[-5,-10],[0,-6.6],[5,-10],[5,-3.3]]):`

Szövegek ábrázolása: Szöveg ábrázolására a `textplot` parancs alkalmas.
`>textplot(sz,opciók);`

`sz`: lista vagy halmaz, amely tartalmazza a szöveg pozícióját és a szöveget

`opciók`: megadhatjuk az igazításokat egy halmazban: `align=t`. A `t` értékei lehetnek: `BELOW`(alatta), `ABOVE`(fölötte), `RIGHT`(jobbra), `LEFT`(balra).

```
>tanfv:=plot(tan(x),x=-Pi..2*Pi,y=-6..6):
>sz:=textplot([seq([i*Pi/2,2,"Szakadás"],i=-1,1)],align=RIGHT):

>display(tanfv,sz):
```

3.4. A plot3d parancs

A 3D grafika legfontosabb függvénye a `plot3d`. Ez a függvény lehetővé teszi, hogy háromdimenzióban felületeket jelenítsünk meg.
A felületek megadása háromféleképpen történhet:

1. $z = f(x, y)$ alakban

2. $r(u,v) = 0$ paraméteres alakban, amelynél a görbe pontjainak x,y,z koordinátáit előállító kifejezéseket vagy függvényeket kell megadni az u és v paraméterekkel. Ezt a fajta ábrázolást úgy képzelhetjük el, hogy az uv sík $[a,b] \times [c,d]$ téglalapját az $(u,v) \to (f(u,v), g(u,v), h(u,v))$ függvénnyel a háromdimenziós térbe képezzük.

3. $f(x,y,z) = 0$ implicit alakban

Maple-ben az első két megadási módnál a `plot3d` függvényt használhatjuk, a felületek implicit megadása esetén a Maple `implicitplot3d` parancsát kell használni. Erről részletesen a 5.1 fejezetben lesz szó.
A `plot3d` használata pontosan olyan egyszerűen történik, mint a `plot` függvényé. Összesen 4 lehetőség közül választhatunk a használata során (amely 4 lehetőség a felületek első két megadási módját fedi le), ezek megadásának módja:

1. `>plot3d(kifejezés,x=a..b,y=c..d,opciók);`

2. `>plot3d(függvény,a..b,c..d,opciók);`

3. `>plot3d([f,g,h],u=a..b,v=c..d,opciók);`

4. `>plot3d([f(u,v),g(u,v),h(u,v)],a..b,c..d,opciók);`

Az első két esetben a felületet Descartes-féle koordinátákkal leírt alakban adjuk meg, míg az utóbbi két esetben a felületet paraméteres alakban adjuk meg a `plot3d` számára.
Az első alaknál az x és az y a `kifejezés` két változója, a második alaknál a `függvény` 2 változója az `a..b` és `c..d` tartományból veszi fel az értékeket. A 3. és 4. változatban paraméteres megadást alkalmazunk, ahol `kg,kh,kl` kifejezések `u,v` paraméterekkel, `fg,fh,fl` pedig függvények.
A `plot3d` opciói a `plot` parancsnál leírtakkal szinte teljesen megegyeznek. Abból adódóan vannak különbségek, hogy itt 3D-s megjelenítés van, értelemszerűen azoknál a paramétereknél, ahol 2 tengelyre vonatkozóan adtuk meg a beállításokat, itt 3 tengelyre kell megadni.
Természetesen vannak kizárólag ennél a függvénynél használható paraméterek, de előtte nézzünk egy pár példát a `plot3d` használatára (12. ábra):

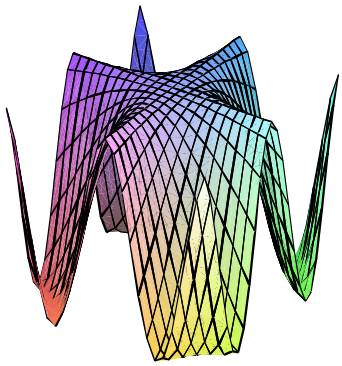
```
>plot3d(-cos(x*y),x=-3..3,y=-2..2):

>f:=(x,y)->cos(x*y):

>plot3d(f,-3..3,-2..2);
```

3.5. A plot3d további opciói

A felület stílusa (style): Lehetőség van a felület stílusának megadására a `style=s` paraméterrel. Az `s` lehetséges értékei:

`POINT`: csak a mintapontokat ábrázolja

`HIDDEN`: olyan mint a `LINE`, csak itt a felületet nem tekinti átlátszónak



12. ábra.

`PATCH`: alapértelmezés

`WIREFRAME`: a mintapontokat vonalakkal köti össze

`CONTOUR`: csak a szintvonalakat ábrázolja, a `contours=n`-nel a szintvonalak számát módosíthatjuk

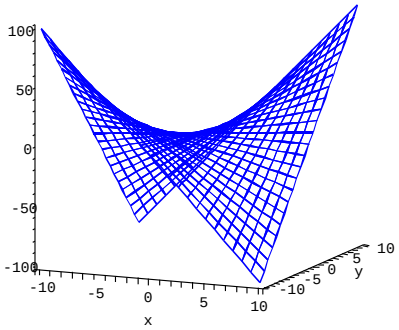
`PATCHNOGRID`: nem rajzol rácshálót a felületre

`PATCHCONTOUR`: `PATCH`+`CONTOUR`

`LINE`: mint a `WIREFRAME`

Példa (13. ábra):

```
>plot3d(x*y,x=-10..10,y=-10..10,style=line,axes=frame,color=blue);
```



13. ábra. Hiperbolikus paraboloid

Árnyékolás, színezés (shading): A felület árnyékolását, színezését is be tudjuk állítani a `shading=s` opcióval. Az `s` lehetséges értékei:

`XYZ`: a pontok x,y és z koordinátáinak összege határozza meg a pont színét (a három tengelyhez külön színskála tartozik)

`XY`: hasonló az `XYZ`-hez, csak az x és y koordinátát veszi figyelembe

`Z`: a pontok z koordinátája szerint történik a színezés

`ZGRAYSCALE`: a z értéktől függő szürkeárnyalatos felület

`ZHUE`: a z értéktől függő HUE színmodell szerinti színezés

Nézőpont (orientation) és vetítés(projection): A `plot3d` az általa kirajzolt függvények képét centrális projekcióval jeleníti meg. Lehetőség van a projekció centrumának megadására is, amit gömbi koordinátákkal kell megadni. Ezt az `orientation=`$[\Theta, \varphi]$ alakban adhatjuk meg, ahol a Θ növelése az ábrát az óramutató járásával ellentétes irányban a z tengely körül forgatja el, a φ a függőleges látószög. Ezeket az értékeket fokban kell megadni. Az alapértelmezett érték 45°. Megadhatjuk a nézőpontnak a felülettől való távolságát is a `projection=r` opcióval, ahol `r` értékei:

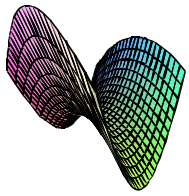
`FISHEYE`: `projection=0`

`NORMAL`: `projection=0`

`ORTHOGONAL`: `projection=1` (alapértelmezett)

Példa (14. ábra)

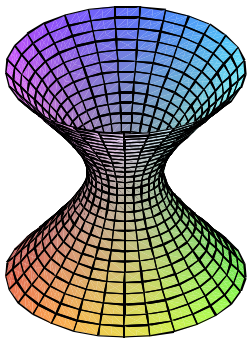
```
>plot3d(x^2/4-y^2/9,x=-10..10,y=-10..10,orientation=[45,125],
projection=FISHEYE);
```



14. ábra.

Rácsméret(grid) és stílus(gridstyle): A `plot3d` a megjelenítés során nem adaptív ábrázolást (2.2. fejezet) alkalmaz (úgy mint a `plot`). Természetesen itt is mintapontok segítségével jeleníti meg a felületet. A mintapontok száma mindkét irányban 25 alapértelmezés szerint. Ezt megváltoztathatjuk a `grid=[a,b]` opcióval, ahol `a` az x, `b` az y irányban jelenti a rácsháló részletességét. Megadhatjuk még a `gridstyle=s` paramétert, ahol `s` értékei lehetnek `rectangular` vagy `triangular`. Ez azt jelenti, hogy a rácsháló négyszögekből vagy háromszögekből álljon. Példa (15. ábra):

```
>plot3d([3*sqrt(1+u^2)*cos(v),3*sqrt(1+u^2)*sin(v),2*u],
v=0..2*Pi,u=-3..3,grid=[30,30]);
```



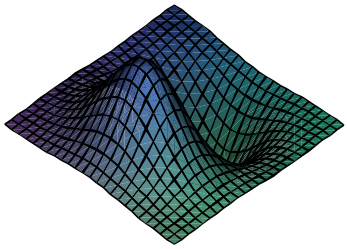
15. ábra. Egyköpenyű hiperboloid

Fények, világítás (light): Lehetőség van a felületet valamilyen fényforrással megvilágítani, ezt kétféle módon tehetjük meg, használhatunk irányított fényforrást (`light`) illetve szórt fényt (`ambientlight`). Ezek megadása:

`light`=$[\varphi, \Theta, r, g, b]$, ahol φ és Θ a fényforrás irányának gömbi koordinátái, `r,g,b`$\in$`[0,1]` a fény színét határozza meg.

`ambientlight=[r,g,b]`, ahol `r,g,b`$\in$`[0,1]`, a fény színe.

```
>plot3d(x*exp(-x^2-y^2),x=-2..2,y=-2..2,light=[90,0,0.4,0.7,
0.8]);
```



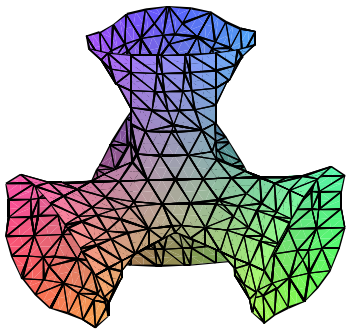
16. ábra.

3.6. Egyéb 3D-s függvények

Számos olyan 2 dimenziós függvény van a Maple-ben, amelynek van 3 dimenziós megfelelője. Ezekre nézünk egy-egy példát.

```
>arrow(seq(<sin(i)+cos(i),sin(i),cos(i)>,i=0..5),labels=[x,y,z],
axes=box);

>fieldplot3d(<x^2,y^2,z^2>,x=-2..2,y=-2..2,z=-2..2,
grid=[6,6,4],axes=box);

>gradplot3d(x^2+y^2+z^2,x=-2..2,y=-2..2,z=-2..2,
grid=[6,6,6],axes=box);

>implicitplot3d(x^3+y^3+z^3+1=(x+y+z+1)^3, x=-2..2,
y=-2..2, z=-2..2, grid=[13,13,13]);
```

(17. ábra, bővebben lásd. 5. fejezet)



17. ábra.

```
>polygonplot3d([[1,1,1],[1,6,6],[6,3,3],[3,1,1]],axes=box);

>textplot3d([5,2,9,"Maple 3D"],align=BELOW,RIGHT,color=red);
```

Kontúrok (szintvonalak) ábrázolása (contourplot3d): A grafikonoknál lehetőség van olyan stílust választani, ahol csak a kontúrok vannak megjelenítve, de van erre külön parancs is.

Kontúrok 2D-s megjelenítése:

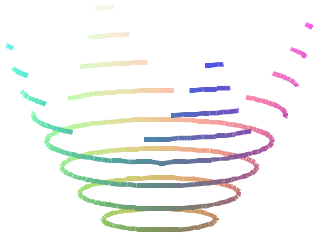
```
>contourplot(kifejezés,x=a..b,y=c..d,opciók);
```

Megadhatjuk az `opciók`-nál a `filled=true` paramétert, melynek hatására kitöltve ábrázolja a kontúrokat. A `coloring=[a,b]` segítségével megadhatjuk a színátmenet két (`a,b`) színét is.

Kontúrok 3D-s megjelenítése:

```
>contourplot3d(kifejezés,x=a..b,y=c..d);

>contourplot(x^2+y^2,x=-5..5,y=-5..5);
```

```
>contourplot3d(x^2+y^2,x=-5..5,y=-5..5,thickness=3);
```
(18. ábra)



18. ábra. Egy elliptikus paraboloid szintvonalai

Mátrixok ábrázolása (matrixplot): A `matrixplot` segítségével lehetőség van egy mátrix értékeinek a szemléltetésére is.
`>matrixplot(M,opciók);`
Az `opciók` a következők lehetnek:

`heights=histogram`: minden elemnek egy-egy oszlop felel meg.

`gap=r`: az oszlopok közötti távolság, ahol $\texttt{r} \in R, r \in [0,1]$

`color=F`: az `F` egy kétváltozós függvény, a színeket határozza meg, ahol a két változó a mátrix sor és oszlopszáma.

Példa (19. ábra):

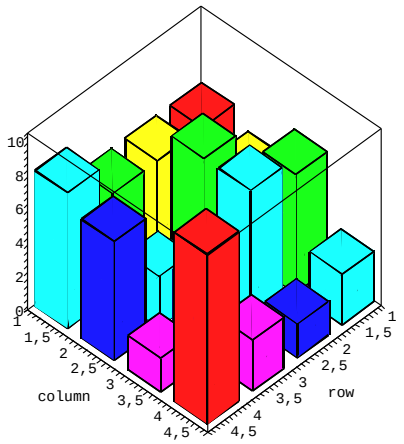
```
>A:=LinearAlgebra:-RandomMatrix(4,4,generator=1..10);

>f:=(i,j)->(i+j);

>matrixplot(A,heights=histogram,gap=0.3,color=f,axes=box);
```

Poliéderek ábrázolása (polyhedraplot):
`>polyhedraplot(P,opciók);`
`P`: halmaz vagy lista, amely a poliéder pontjait tartalmazza
Az `opciók` lehetnek:

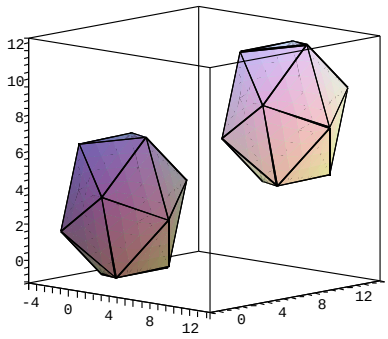
`polyscale=n`: az `n` egész szám, alapértelmezésben 1, a nagyítás mértéke.

`polytype=s`: az `s` lehet `tetrahedron`, `icosahedron`, `dodecahedron`, `hexahedron`, `octahedron` (az 5 szabályos poliéder)

Példa (20. ábra):

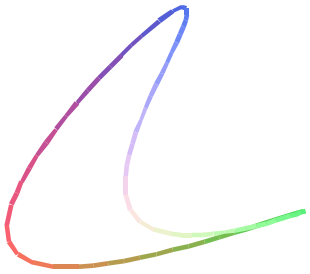


19. ábra.



20. ábra. Két ikozaéder

Térgörbék rajzolása (spacecurve): A számítógépes algebrai rendszerek egyik nagy előnye, hogy képes háromdimenzióban görbét ábrázolni. A görbe valós idejű forgatása nagy segítség a görbe alakjának elképzeléséhez. A `spacecurve` parancs fontos a számunkra, hiszen nagy szerepe lesz a másodrendű felületek metszésvonalainak és áthatásának szemléltetésénél. A háromdimenziós görbét a `spacecurve` függvénynek paraméteres alakban kell megadni.
Alakja:
`>spacecurve([f(t),g(t),h(t)],t=a..b,opciók);`
Az `f`, `g` és `h` a görbe egy adott `t` értékéhez tartozó `x`,`y` és `z` koordinátáit határozzák meg. Az `opciók` azonosak a `plot3d` opcióival, kivéve a `grid`-et, ezt nem alkalmazhatjuk. Nézzünk egy példát (21. ábra):
`>spacecurve([sin(t)*cos(t),sin(t),cos(t)],t=0..2*Pi,thickness=3);`
Amennyiben több görbét szeretnénk megadni, azokat halmazban kell megadni.
A `t` tartományát megadhatjuk a listán belül is. Ez akkor hasznos, ha több görbe esetén a `t` tartománya nem egyezik meg.
A `numpoints` alapértelmezett értéke 50. A függvények helyett pontokat is megadhatunk a listában, ezeket a megadott sorrendben egymásután szakaszokkal köti



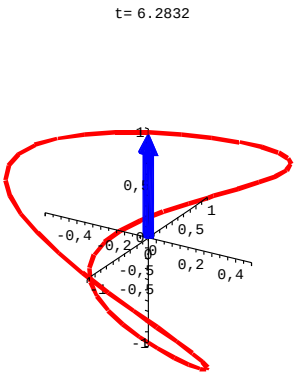
21. ábra.

össze.

```
>spacecurve([[0,0,0],[5,0,0],[5,5,5],[2,5,5],[0,2,3],[0,0,0]],
thickness=3,axes=box):
```

Térgörbék szemléltetésénél jól használhatjuk az `arrow` függvényt. Nézzünk erre egy példát (22. ábra):

```
>f:=t->[sin(t)*cos(t),sin(t),cos(t)]:

>nyil:=animate(arrow,[f(t),color=blue],t=0..2*Pi):

>gorbe:=spacecurve(f(t),t=0..2*Pi,axes=normal,color=red,
thickness=3):

>display([gorbe,nyil]);
```

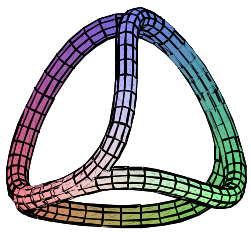


22. ábra.

Egy ilyen animáció segítségével könnyen megérthető a kapcsolat a vektorok és a térgörbék között.

A tubeplot használata: Némely esetben a `spacecurve` használata nem elég szemléletes. Ilyenkor használhatjuk a `tubeplot` függvényt. Ez hasonlóképpen működik mint a `spacecurve`, de itt a görbe vonalát körbeveszi egy adott sugarú hengeres felülettel. A sugár alapértelmezésben 1, ezt a `radius=n` ($n \in R$) értékkel módosíthatjuk.
Alakja:
`>tubeplot([f(t),g(t),h(t)],t=a..b,opciók);`
A `numpoints` alapértelmezett értéke 50, illetve megadható még a `tubepoints` paraméter (alapértelmezett értéke 10), ami a görbét körülvevő henger részletességét jelenti. Példa (23. ábra):

```
>tubeplot({[sin(t),cos(t),sin(t)*cos(t)],
[cos(t),sin(t)*cos(t),sin(t)]},t=0..2*Pi,radius=0.1);
```



23. ábra.

4. Koordináta-rendszerek Maple-ben

A Maple-ben lehetőség van a Descartes-féle (Cartesian) koordináta-rendszertől különböző koordináta-rendszerek használatára is. Számos koordináta-rendszerben lehetőségünk van dolgozni (pl. `bispherical`, `conical`, `paraboloidal`, stb.), mi most a polár mellett a szférikus (vagy gömbi) és henger koordináta-rendszereket nézzük meg, ugyanis e két utóbbi fontos a másodrendű felületek ábrázolásánál is.

4.1. A polár koordináta-rendszer

A `plot` utasítás paraméterei között szerepel a `coords` is, ami lehetővé teszi, hogy a Descartes-féle koordináta-rendszertől különböző koordináta-rendszert használjunk. Amennyiben a `coords=polar` -t választjuk, akkor a pontokat a Maple az (r, Θ) alakban fogja ábrázolni, ahol r ($r \geq 0$) a pont origótól való távolsága, Θ pedig az x tengely pozitív végével az óramutató járásával ellentétes irányban a ponttal bezárt szög.
A Maple-ben az ábrázoláshoz meg kell adni r függvényét, melyben a Θ a paraméter és Θ értelmezési tartományát. A szintaktika:

```
>plot(r(t),t=a..b,coords=polar);
```

Például:

```
>plot(cos(2*t),t=0..2*Pi,coords=polar):
```

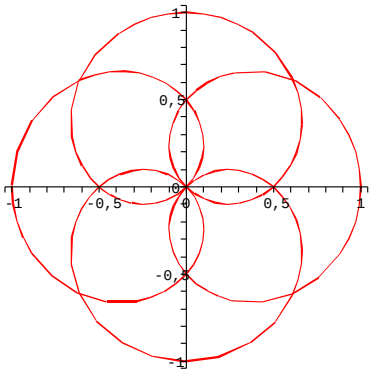
A paraméteres megadáshoz a szintaxis:

```
>plot([r(t),theta(t),t=a..b],coords=polar);
```

Például (24. ábra):

```
>plot([cos(2*t),3*t,t=0..2*Pi],coords=polar);
```

A másik lehetőség polár koordináta-rendszerben történő ábrázoláshoz a `polarplot`



24. ábra.

parancs használata. Paramétereinek megadása az előbbiekkel megegyezik, egyedül a `coords=polar` részt kell elhagyni. Például:

```
>polarplot([sin(t),cos(4*t),t=0..2*Pi]):
```

4.2. A henger koordináta-rendszer

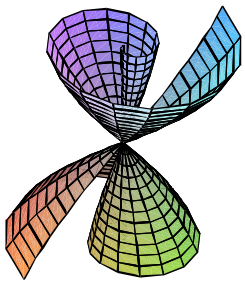
A henger koordináta-rendszerben történő ábrázoláshoz a `coords=cylindrical` paramétert kell megadni. Az ilyen rendszerben történő ábrázolást jól használhatjuk, ha másodrendű vagy körszimmetrikus felületeket részletesen (pontosan) szeretnénk megjeleníteni.
Ebben a rendszerben egy P pontot az (r, Θ, m) hármas definiál, ahol $r(r \geq 0)$ a távolság a z tengely és a P között, Θ a szög az x tengely pozitív vége és a sugár xy síkra eső vetülete között, az m az xy sík és a P közötti előjeles távolság. Megadása:

```
>plot3d(r(t,m),t=a..b,m=c..d,coords=cylindrical);
```

Például (25. ábra):

```
>plot3d(sin(m)*t,t=0..3*Pi,m=0..1,coords=cylindrical);
```

A paraméteres megadás:



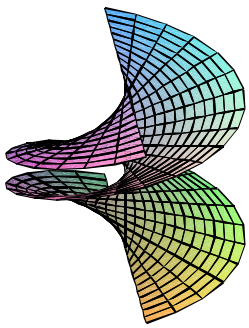
25. ábra.

```
>plot3d([r(u,v),t(u,v),m(u,v)],u=a..b,v=c..d,coords=cylindrical);
```

Például (26. ábra):

```
>plot3d([u,v,u+v],u=-1..1,v=-Pi..Pi,coords=cylindrical);
```

Lehetőség van arra is, hogy ne az r-et, hanem az m-et fejezzük ki az r és a Θ függ-



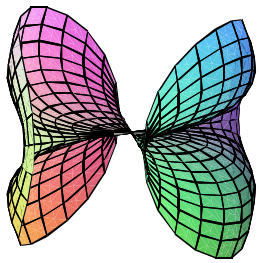
26. ábra.

vényében, ennek megadása:

```
>plot3d([r,t,m(r,t)],r=a..b,t=c..d,coords=cylindrical);
```

Például (27. ábra):

```
>plot3d([sin(2*r),cos(t),r^2*sin(t)],r=-1..1,t=-Pi..Pi,
coords=cylindrical);
```



27. ábra.

A másik lehetőség a henger koordináta-rendszer használatára a `cylinderplot` parancs alkalmazása. Ennek használatát a Maple készítői azonban nem javasolják, helyette a `plot3d` használata javasolt.

4.3. A gömbi koordináta-rendszer

A gömbi koordináta-rendszerben történő ábrázoláshoz a `coords=spherical` paramétert kell megadnunk. A gömbi koordinátákat, csakúgy mint a henger koordinátákat is térbeli alakzatok megjelenítésénél használhatjuk.
Egy pontot a (r, Θ, φ) hármas határoz meg, ahol az r(sugár) a pont origótól való távolsága (negatív is lehet), a Θ ugyanazt jelenti mint a polár koordináta-rendszerben, a φ a z tengely pozitív végével és a sugárral bezárt szög. Megadása:

```
>plot3d(r(t,f),t=a..b,f=c..d,coords=spherical);
```

Például (28. ábra):

```
>plot3d(sin(t)+cos(2*f),t=0..2*Pi,f=0..Pi,coords=spherical);
```

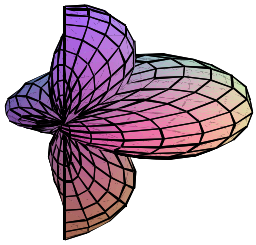
A paraméteres alakban történő megadás:

```
>plot3d([r(u,v),t(u,v),f(u,v)],u=a..b,v=c..d,coords=spherical);
```

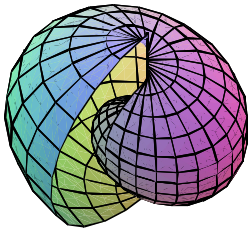
Például (29. ábra):

```
>plot3d([u+v,u,v],u=0..2*Pi,v=0..2*Pi,coords=spherical);
```

A másik lehetőség a gömbi koordináták használatához a `sphereplot` parancs, de ezt a parancsot sem javasolják már használni a Maple készítői.



28. ábra.



29. ábra.

5. Másodrendű felületek Maple-ben

A másodrendű felületeket leíró általános egyenlet alakja:

$$a \cdot x^2 + b \cdot y^2 + c \cdot z^2 + 2 \cdot f \cdot y \cdot z + 2 \cdot g \cdot z \cdot x + 2 \cdot h \cdot x \cdot y + 2 \cdot p \cdot x + 2 \cdot q \cdot y + 2 \cdot r \cdot z + d = 0$$

A Maple képes arra, hogy ezeket a felületeket megjelenítse és valós időben lehetőség van a forgatásukra, mozgatásukra. A másodrendű felületek egy rövid ismertetése után megnézzük az ábrázolásukat Maple-ben, majd a síkmetszeteiket illetve az áthatásukat.
A másodrendű felületeket minden sík egy nemelfajult vagy elfajult, valós vagy képzetes másodrendű görbében metszi.
A nemelfajult másodrendű felületek affin szempontból a következők lehetnek: ellipszoid, elliptikus kúp, elliptikus henger, elliptikus paraboloid, hiperbolikus henger, hiperbolikus paraboloid, egyköpenyű hiperboloid, kétköpenyű hiperboloid, parabolikus henger.

5.1. Az implicitplot3d parancs

Az előbbi felületek ábrázolásához (kezdetben) az `implicitplot3d` parancsot használjuk, amelynek egy egyenletet kell megadni paraméterként, és egy numerikus algoritmust (úgynevezett háromszögelést) használ a felület ábrázolásához. Alakja:

```
>implicitplot3d(e,x=a..b,y=c..d,opciók);
```

A `gridstyle` paraméter kivételével az `opciók` megegyeznek a `plo3d`-vel. A `gridstyle` az `implicitplot3d` esetén rögzített, `triangular`.
Az `implicitplot3d` segítségével a felületeket elég durván tudjuk ábrázolni (mint azt majd most látni fogjuk, ez a parancs a gyors szemléltetéshez jó), ha szebb, részletesebb ábrákat szeretnénk, akkor valamilyen átalakításra vagy trükkre van szükség, ezeket majd az 5.3-5.6. fejezetekben részletezzük.
Most azonban nézzük meg az egyes másodrendű felületeket részletesebben.

5.2. Másodrendű felületek

5.2.1. Ellipszoid

Szokás háromtengelyű ellipszoidnak is nevezni. Descartes-féle koordinátákkal megadott egyenlete:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$$

(ahol a féltengelyek hossza: a, b, c). A gömbi koordinátákkal megadott egyenlete:

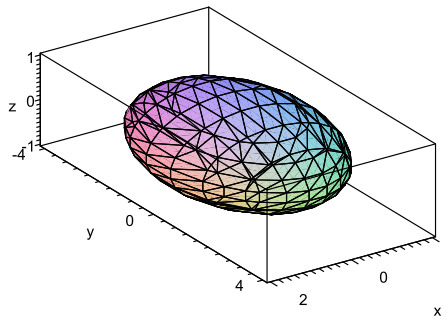
$$\frac{r^2 \cdot cos(\Theta)^2 \cdot sin(\varphi)^2}{a^2} + \frac{r^2 \cdot sin(\Theta)^2 \cdot sin(\varphi)^2}{b^2} + \frac{r^2 \cdot cos(\varphi)^2}{c^2} = 1$$

Ha 2 tengely hossza megegyezik, akkor a neve forgási ellipszoid (vagy szferoid), ha $c < a$, akkor lapított ellipszoid, ha $c > a$, akkor nyújtott ellipszoid. Ha mindhárom tengely hossza egyenlő, akkor gömbnek nevezzük.
Az ellipszoid paraméteres egyenlete:

$$x = a \cdot cos(\Theta) \cdot sin(\varphi),\ y = b \cdot sin(\Theta) \cdot sin(\varphi),\ z = c \cdot cos(\varphi)$$

ahol $\Theta \in [0, 2\Pi)$ és $\varphi \in [0, \Pi]$.
Példa(30. ábra):

```
>setoptions3d(scaling=constrained,axes=box):

>ellipszoid:=x^2/4+y^2/9+z^2=1:

>implicitplot3d(ellipszoid,x=-2..2,y=-4..4,z=-1..1);
```



30. ábra. Ellipszoid

5.2.2. Elliptikus kúp

Az elliptikus kúp egyenlete:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = z^2$$

A paraméteres egyenlete:

$$x = a \cdot \tfrac{h-u}{h} \cdot cos(v),\ y = b \cdot \tfrac{h-u}{h} \cdot sin(v),\ z = u$$

ahol h a kúp magassága, a a nagytengely, b a kistengely fele és $v \in [0, 2\Pi)$, $u \in [0, h]$.
Példa (31. ábra):

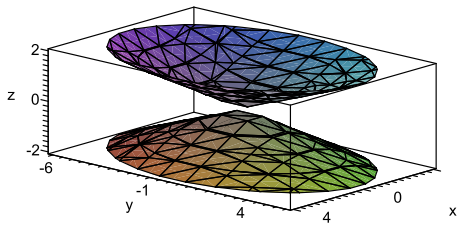
```
>ellkup:=z^2=x^2/4+y^2/9:

>implicitplot3d(ellkup,x=-4..4,y=-6..6,z=-2..2);
```

Ha megnézzük az elliptikus kúp képét, láthatjuk hogy az ábrán nincs rajta a kúp csúcsa. Ez azért van, mert alapértelmezésben a `grid` paraméter mindhárom tartományban 10-es értékkel dolgozik, ezért nincs rajta a csúcs az ábrán. Ha a `grid` paraméternek páratlan értéket adunk meg mindhárom irányban, akkor már ott lesz a kúp csúcsa is.

5.2.3. Elliptikus henger

Általános egyenlete:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$



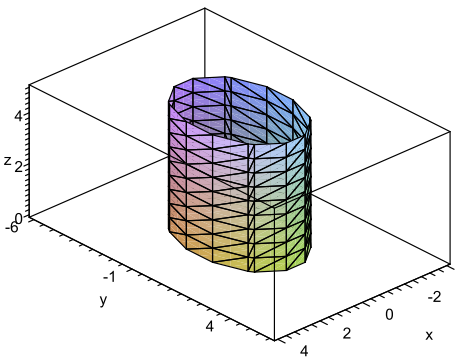
31. ábra. Elliptikus kúp

Az elliptikus henger olyan henger, amelynek alapgörbéje ellipszis. A paraméteres egyenlete:

$$x = a \cdot cos(u),\ y = b \cdot sin(u),\ z = v$$

ahol h a henger magassága, a a nagytengely, b a kistengely hosszának fele és $u \in [0, 2\Pi)$, $v \in [0, h]$.
Példa (32. ábra):

```
>ellhenger:=x^2/4+y^2/9=1:
```



32. ábra. Elliptikus henger

5.2.4. Elliptikus paraboloid

Általános egyenlete:

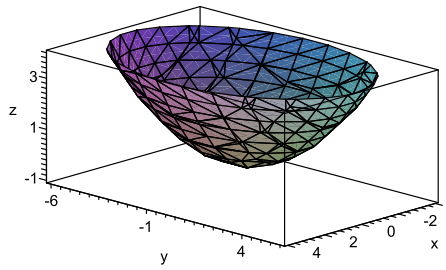
$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = z$$

Paraméteres egyenlete:

$$x = a \cdot \sqrt{u} \cdot cos(v),\ y = b \cdot \sqrt{u} \cdot sin(v),\ z = u$$

ahol a, b a tengelyek hosszának fele, h a magasság, $v \in [0, 2\Pi)$, $u \in [0, h]$.
Példa (33. ábra):

```
>ellpar:=z=x^2/4+y^2/9;

>implicitplot3d(ellpar,x=-4..4,y=-6..6,z=-1..4):
```



33. ábra. Elliptikus paraboloid

5.2.5. Hiperbolikus henger

Általános egyenlete:

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = -1$$

Paraméteres egyenlete:

$$x = a \cdot sinh(u),\ y = b \cdot cosh(u),\ z = v$$

ahol $u, v \in (-\infty, \infty)$.
Példa (34. ábra):

```
>hiphenger:=x^2/4-y^2/9=-1:

>implicitplot3d(hiphenger,x=-4..4,y=-6..6,z=0..5);
```

5.2.6. Hiperbolikus paraboloid

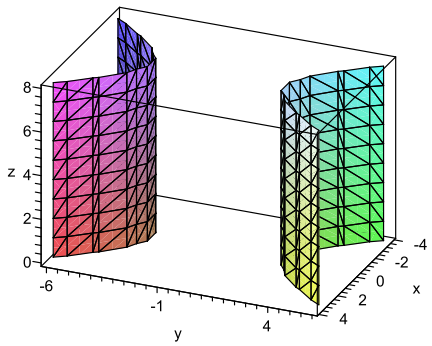
Általános egyenlete:

$$\frac{y^2}{a^2} - \frac{x^2}{b^2} = z$$

Egy alternatív alakja:

$$z = x \cdot y$$

Paraméteres egyenlete:

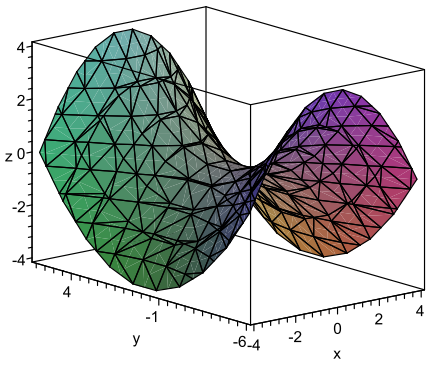


34. ábra. Hiperbolikus henger

$$x(u,v) = u,\ y(u,v) = v,\ z(u,v) = u \cdot v$$

ahol $u, v \in (-\infty, \infty)$.
Példa (35. ábra):

```
>hippar:=z=y^2/9-x^2/4:

>implicitplot3d(hippar,x=-4..4,y=-6..6,z=-4..4);
```



35. ábra. Hiperbolikus paraboloid

5.2.7. Egyköpenyű hiperboloid

Az egyköpenyű hiperboloid általános egyenlete:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} - \frac{z^2}{c^2} = 1$$

A hiperboloidoknál megkülönböztetünk egy és kétköpenyűt.
Abban az esetben, ha $a = b$, akkor forgáshiperboloidról beszélünk. Az egyköpenyű

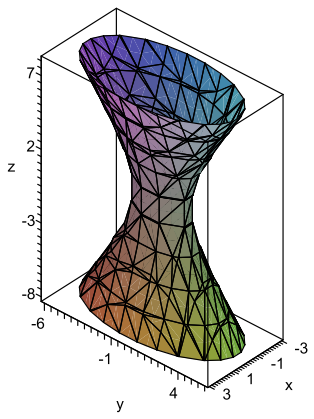
forgáshiperboloid egy hiperbola olyan forgásfelülete, melynek tengelye a hiperbola fókuszpontjait összekötő szakasz felezőmerőlegese (ezt nevezik képzetes tengelynek).
A kétköpenyű forgáshiperboloid szintén egy hiperbola forgásfelülete, de ott a forgástengely a fókuszpontokat összekötő egyenes (ez a valós tengely).
Paraméteres egyenletei:

1.
$$x = a \cdot \sqrt{1+u^2} \cdot cos(v)$$
$$y = b \cdot \sqrt{1+u^2} \cdot sin(v)$$
$$z = c \cdot u$$
ahol $v \in [0, 2\Pi]$ és $u \in (-\infty, \infty)$

2.
$$x(u,v) = a \cdot (cos(u) \mp v \cdot sin(u))$$
$$y(u,v) = b \cdot (sin(u) \pm v \cdot cos(u))$$
$$z(u,v) = \pm c \cdot v$$
ahol $u \in [0, 2\Pi]$ és $v \in (-\infty, \infty)$

3.
$$x(u,v) = a \cdot cosh(v) \cdot cos(u)$$
$$y(u,v) = b \cdot cosh(v) \cdot sin(u)$$
$$z(u,v) = c \cdot sinh(v)$$
ahol $u \in [0, 2\Pi]$ és $v \in (-\infty, \infty)$

Példa (36. ábra):
```
>hipegykop:=x^2+y^2/4-z^2/9=1:

>implicitplot3d(hipegykop,x=-3..3,y=-6..6,z=-8..8);
```



36. ábra. Egyköpenyű hiperboloid

5.2.8. Kétköpenyű hiperboloid

A kétköpenyű hiperboloid egyenlete:

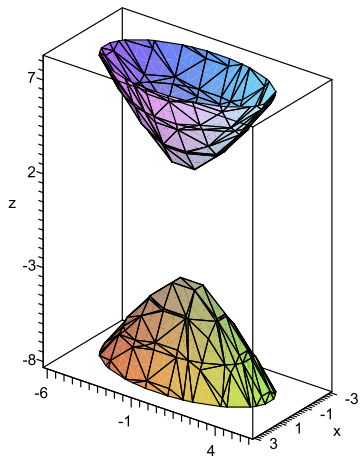
$$\frac{x^2}{a^2} + \frac{y^2}{b^2} - \frac{z^2}{c^2} = -1$$

Paraméteres egyenlete:

$$x = a \cdot sinh(u) \cdot cos(v)\ y = b \cdot sinh(u) \cdot sin(v)\ z = c \cdot cosh(u)$$

ahol $u \in (-\infty, \infty)$ és $v \in [0, 2\Pi]$
Példa (37. ábra):

```
>hipketkop:=x^2+y^2/4-z^2/9=-1:

>implicitplot3d(hipketkop,x=-3..3,y=-6..6,z=-8..8);
```



37. ábra. Kétköpenyű hiperboloid

5.2.9. Parabolikus henger

Általános egyenlete:

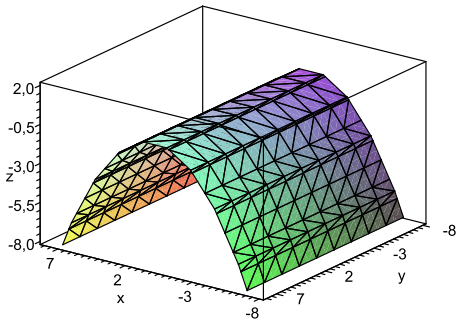
$$x^2 + 2 \cdot r \cdot z = 0$$

Példa (38. ábra):

```
>parhen:=x^2+6*z=0:

>implicitplot3d(parhen,x=-8..8,y=-8..8,z=-8..2);
```

5.3. Paraboloidok ábrázolása

Az elliptikus és hiperbolikus paraboloidákat egyszerűen ábrázolhatjuk a `plot3d` parancs segítségével is, hiszen az általános egyenletük a `z` változó szerint van felírva. A `plot3d` segítségével így már részletesebb, szebb ábrát kapunk.



38. ábra. Parabolikus henger

```
>plot3d(rhs(ellpar),x=-4..4,y=-6..6):

>plot3d(rhs(hippar),x=-4..4,y=-6..6):
```

5.4. Elliptikus kúpok ábrázolása

Elliptikus kúpok jó minőségű megjelenítéséhez használhatjuk a paraméteres egyenletét vagy áttérhetünk henger koordináta-rendszerre. Mi most ez utóbbit nézzük meg. Az áttéréshez az $x = r \cdot cos(\Theta)$ és $y = r \cdot sin(\Theta)$ átalakításokat kell felhasználni (39. ábra):

```
>ellkup:=x^2/4+y^2/9-z^2=0:

>x:=r*cos(t):y:=r*sin(t):

>collect(simplify(ellkup),r);
```

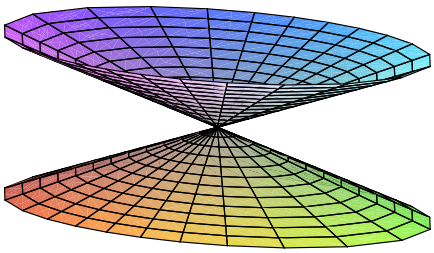
$$\left(\frac{5}{36}cos(t)^2 + \frac{1}{9}\right)r^2 - z^2 = 0$$

```
>r:=sqrt(simplify(z^2/((lhs(%)+z^2)/r^2))):

>plot3d(r,t=0..2*Pi,z=-2..2,coords=cylindrical,scaling=constrained);
```

5.5. Ellipszoidok ábrázolása

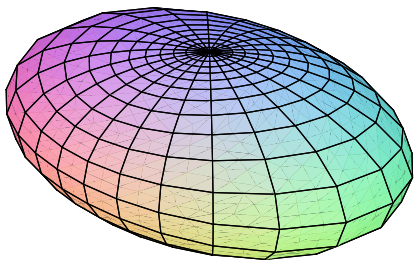
Az ellipszoidok ábrázolása történhet gömbi és henger koordináta-rendszer használatával is. Mi most csak a gömbibe való áttérést nézzük meg (a henger az előzőhöz hasonló módon történik).
Az átírást a következőképpen tehetjük meg (40. ábra):

$$\frac{r^2cos(\Theta)^2sin(\varphi)^2}{a^2} + \frac{r^2sin(\Theta)^2sin(\varphi)^2}{b^2} + \frac{r^2cos(\varphi)^2}{c^2} = 1$$



39. ábra.

```
>ellipszoid:=(r^2*cos(t)^2*sin(f)^2)/4+(r^2*sin(t)^2*sin(f)^2)/9+
+(r^2*cos(f)^2)=1;
>collect(ellipszoid,r):

>r:=sqrt(1/(lhs(%)/r^2)):

>plot3d(r,t=0..2*Pi,f=0..Pi,coords=spherical,scaling=constrained);
```



40. ábra.

5.6. Hiperboloidok ábrázolása

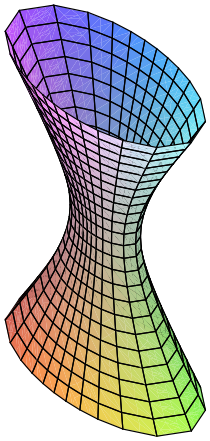
A hiperboloidok ábrázolásánál henger koordináta-rendszerbe kell áttérni ($x = r \cdot cos(\Theta)$, $y = r \cdot sin(\Theta)$).
Egyköpenyű hiperboloid (41. ábra):

```
>hipegykop:=x^2+y^2/4-z^2/9=1:
```

```
>x:=r*cos(t):y:=r*sin(t):

>collect(hipegykop,r);
```

$$\left(cos(t)^2+\frac{1}{4}sin(t)^2\right)r^2-\frac{1}{9}z^2=1$$

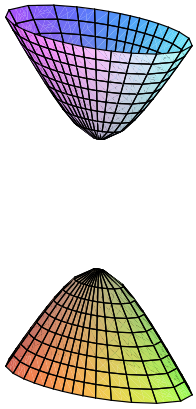
```
>r:=sqrt((1+1/9*z^2)/(cos(t)^2+1/4*sin(t)^2)):

>plot3d(r,t=0..2*Pi,z=-8..8,coords=cylindrical,scaling=constrained);
```



41. ábra.

Kétköpenyű hiperboloid (42. ábra):

```
>hipketkop:=x^2+y^2/4-z^2/9=-1:

>x:=r*cos(t):y:=r*sin(t):

>collect(hipketkop,r):

>r:=sqrt((-1+1/9*z^2)/(cos(t)^2+1/4*sin(t)^2)):

>plot3d(r,t=0..2*Pi,z=-8..8,coords=cylindrical,scaling=constrained);
```



42. ábra.

6. Animációk készítése Maple-ben

A Maple-ben lehetőség van animációk készítésére is. A Maple bármilyen két vagy háromdimenziós grafikus parancsot tud animálni. Három parancs áll rendelkezésünkre animációhoz:

- `animate`
- `animatecurve`
- `animate3d`

Az újabb Maple verziókban az `animate3d` parancs helyett az `animate` parancs használatát ajánlják. Az animáció képkockákból épül fel, melyeket a Maple egymás után vetít le. Az animációkat az eszköztáron lévő (az animációra történő kattintás után) animációs gombokkal tudjuk vezérelni.
Mivel a szakdolgozat alapvetően a felületábrázolásra helyezi a hangsúlyt, az egyváltozós függvények animációjáról csak röviden lesz szó.

6.1. Az animate parancs 2D-ben

Alakja:

```
>animate(parancs,[argumentumok],t=a..b);
```

A `parancs` egy tetszőleges 2D-s vagy 3D-s grafikus parancs lehet. Az `argumentumok` a `parancs`-hoz tartozó paraméterlista, a `t=a..b` az animációban szereplő animálandó változó tartománya, $a, b \in R$ (ilyen változókból természetesen többet is megadhatunk itt).
Alapértelmezésben az `animate` 25 képkockát generál, amit a `t` tartományban egyenletesen oszt el. Természetesen az animációnak csak úgy van értelme, ha a `t`-t felhasználjuk a grafikus parancsban. Például:

```
>animate(plot,[sin(x),x=-t..t],t=-Pi..Pi);
```

A generált képkockák számát az animációs eszköztáron utólag is tudjuk módosítani, de a `frames=n` paraméterrel is megváltoztathatjuk.

```
>animate(plot,[sin(x)/t,x=-t..t,coords=polar],t=-2*Pi..2*Pi,
frames=50);
```

Lehetőség van az animáció hátterében egy ábra elhelyezésére a `background=P` paraméterrel, ahol `P` egy `PLOT` vagy egy `PLOT3D` objektum lehet.

```
>curve:=implicitplot3d(x^3+y^2,x=-3..1,y=-4..4,color=blue):
>animate(implicitplot,[x^3-A*y+y^2,x=-3..1,y=-4..4],A=-2..2,
background=curve):
```

Egy másik példa:

```
>animate(plot,[1+a*cos(t),t=0..2*Pi,coords=polar],a=-2..2):
```

6.2. Az animatecurve parancs

Az `animatecurve` függvény segítségével valós függvényeket szemléltethetünk, animálhatunk kétdimenzióban. Ezek a valós függvények megadhatók kifejezéssel, függvénnyel vagy paraméteres alakban. A generált képkockák száma alapértelmezésben 16, de

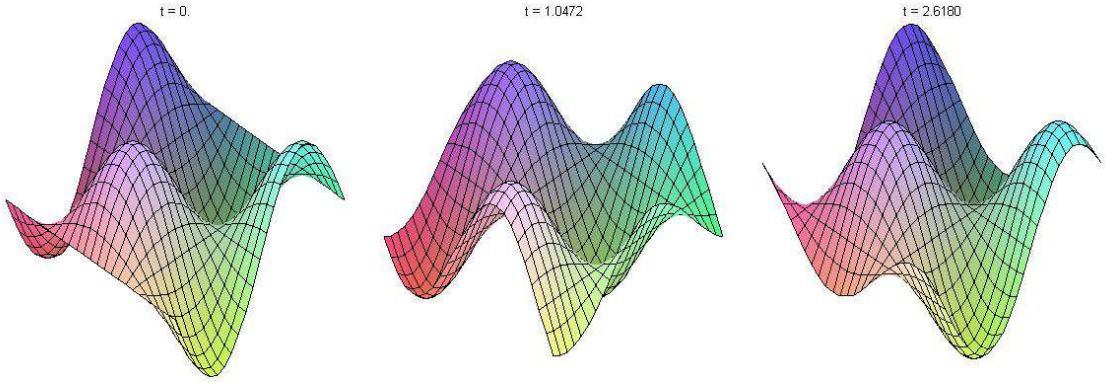
itt is változtatható. A megadható egyéb paraméterek nagyjából megegyeznek a `plot` paramétereivel.

```
>f:= x->sin(x):

>animatecurve(f,-Pi..Pi,coords=polar):
```

6.3. Az animate parancs 3D-ben

A háromdimenziós parancsok esetén is teljesen hasonlóan megy az animáció, mint kétdimenzió esetén. Nézzünk egy pár példát :

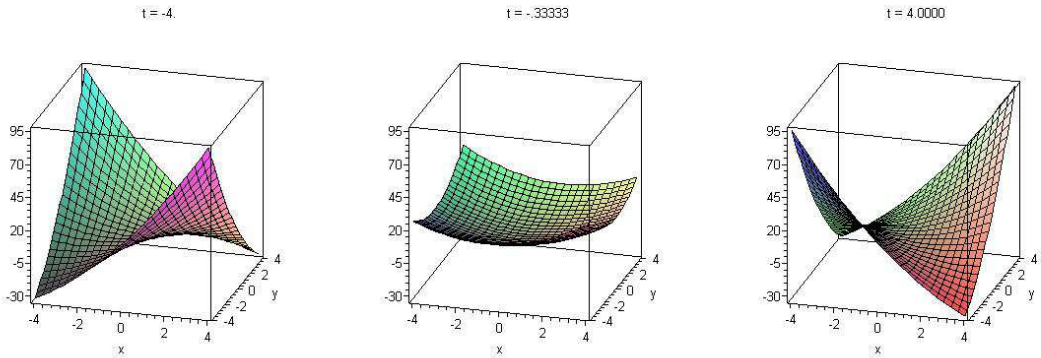
```
>animate(plot3d,[sin(x-t)*cos(y-t),x=0..2*Pi,y=0..2*Pi],t=0..2*Pi);
```



43. ábra.

```
>animate(plot3d,[x^2+t*x*y+y^2,x=-4..4,y=-4..4],t=-4..4,axes=box);
```

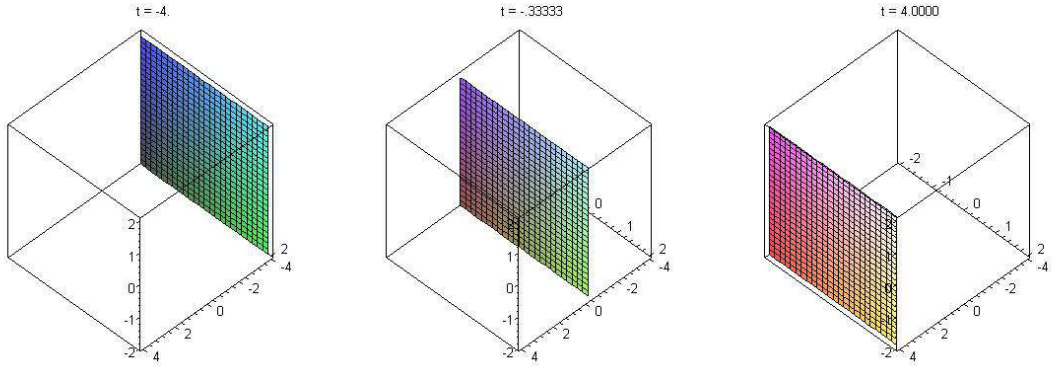
Az animálandó függvényt paraméteres alakban is megadhatjuk. A következő animáció



44. ábra.

egy függőleges síkot mozgat az x tengely mentén.

```
>animate(plot3d,[[t,y,z],y=-2..2,z=-2..2],t=-4..4,axes=box);
```



45. ábra.

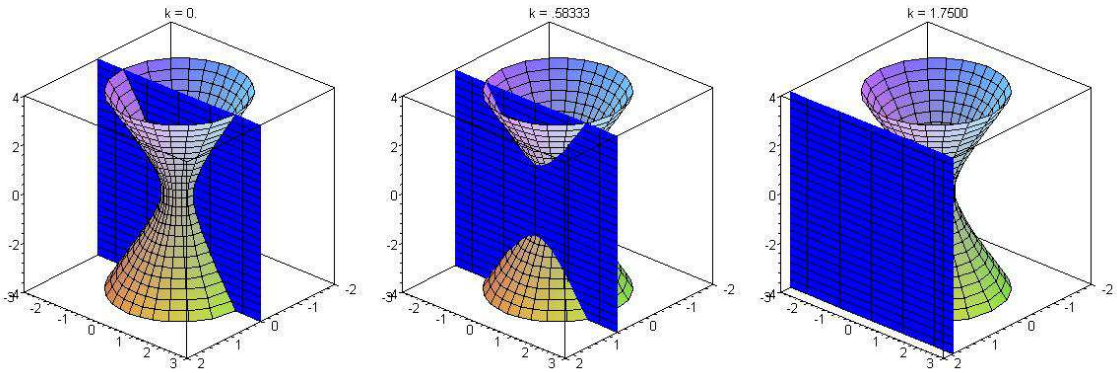
6.4. Animáció hiperboloidokkal

A másodrendű felületek alakjának megértésében segít, ha megvizsgáljuk a felület metszésvonalát a felület tengelyével párhuzamos illetve rá merőleges síkok mentén.

6.4.1. Egyköpenyű hiperboloid

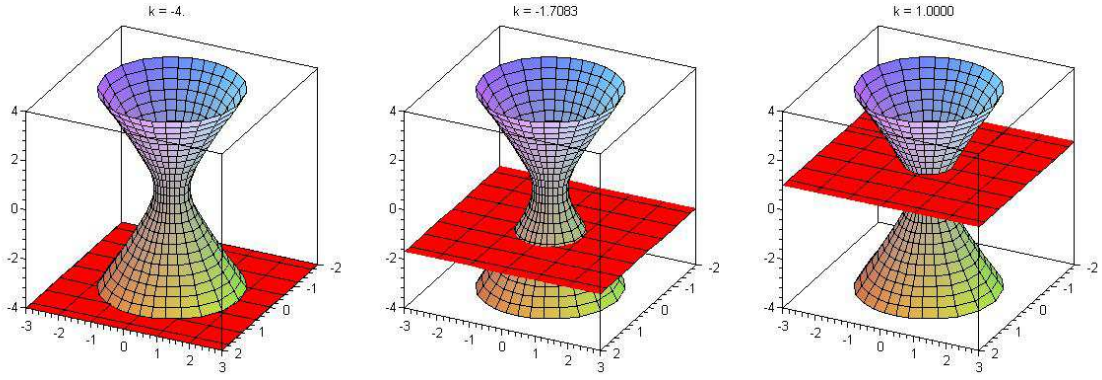
A következő animációban egy egyköpenyű hiperboloid metszésvonalát szemléltetjük függőleges és vízszintes síkokkal.

```
>r:=sqrt((z^2+1)/(9*cos(t)^2+4*sin(t)^2)):

>hipegykop:=plot3d(r,t=0..2*Pi,z=-4..4,coords=cylindrical,axes=box):

>fuggsikok:=animate(plot3d,[[k,y,z],y=-14..14,z=-4..4,color=blue],
k=0..7):

>vizsikok:=animate(plot3d,[[x,y,k],x=-7..7,y=-12..12,color=red],
k=-4..1):

>display([hipegykop,fuggsikok]); (46. ábra)
```



46. ábra. Egyköpenyű hiperboloid függőleges síkokkal

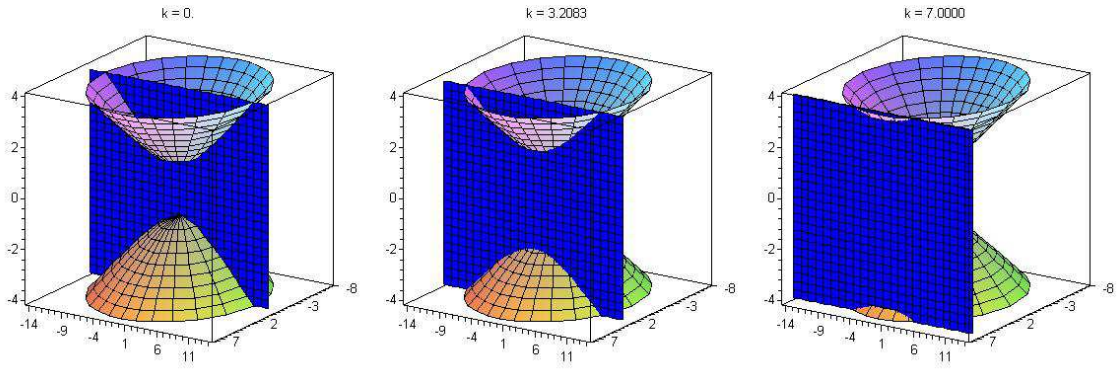
```
>display([hipegykop,vizsikok]); (47. ábra)
```



47. ábra. Egyköpenyű hiperboloid vízszintes síkokkal

6.4.2. Kétköpenyű hiperboloid

```
>r:=6*sqrt((z^2-1)/(9*cos(t)^2+4*sin(t)^2)):

>hipketkop:=plot3d(r,t=0..2*Pi,z=-4..4,coords=cylindrical,axes=box):

>display([hipketkop,fuggsikok]); (48. ábra)
```

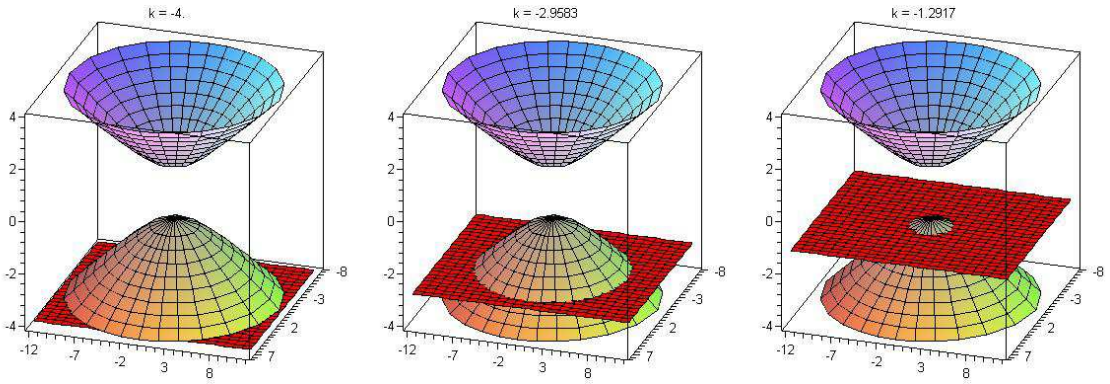


48. ábra. Kétköpenyű hiperboloid függőleges síkokkal

```
>display([hipketkop,vizsikok]); (49. ábra)
```

6.5. Animáció paraboloidokkal

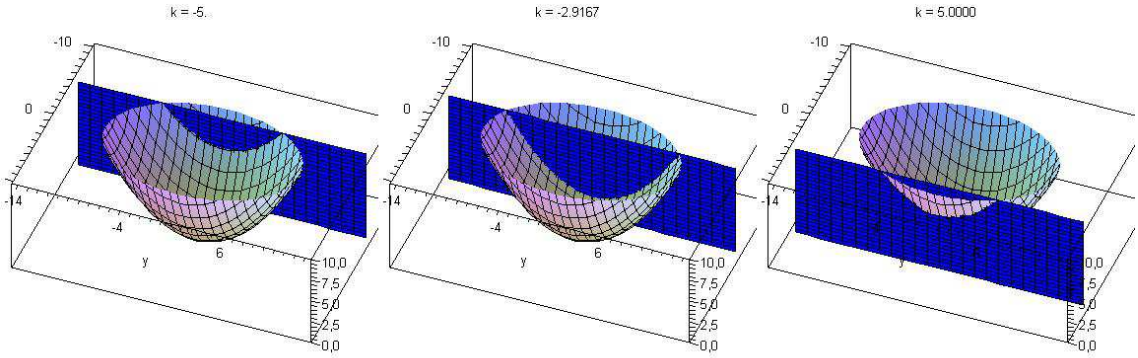
A következő animációk az elliptikus és a hiperbolikus paraboloidok metszésvonalait szemléltetik.



49. ábra. Kétköpenyű hiperboloid vízszintes síkokkal

6.5.1. Elliptikus paraboloid

```
>ellpar:=plot3d(x^2/4+y^2/9,x=-10..10,y=-10..10,axes=box,view=0..10,
scaling=constrained):

>fuggsikok:=animate(plot3d,[[k,y,z],y=-14..14,z=0..10,color=blue],
k=-5..5):

>vizsikok:=animate(plot3d,[[x,y,k],x=-8..8,y=-10..10,color=red],
k=0..10):

>display([ellpar,fuggsikok]); (50. ábra)
```

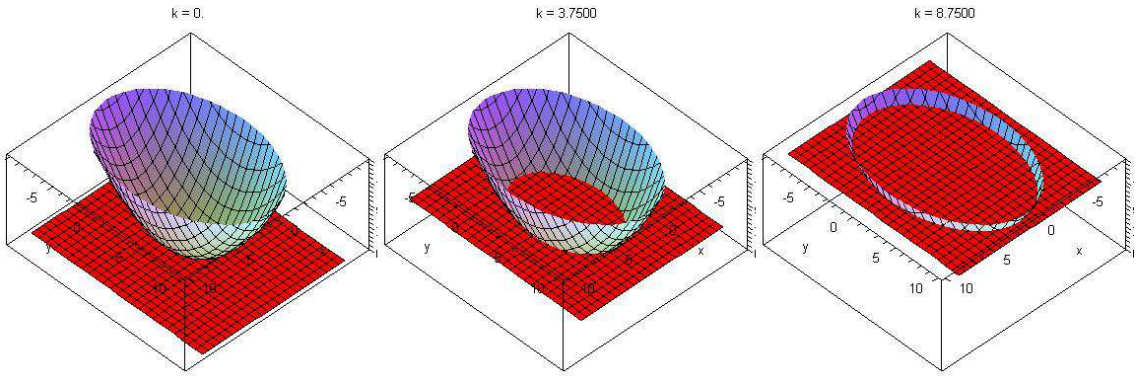


50. ábra. Elliptikus paraboloid függőleges síkokkal

```
>display([ellpar,vizsikok]); (51. ábra)
```

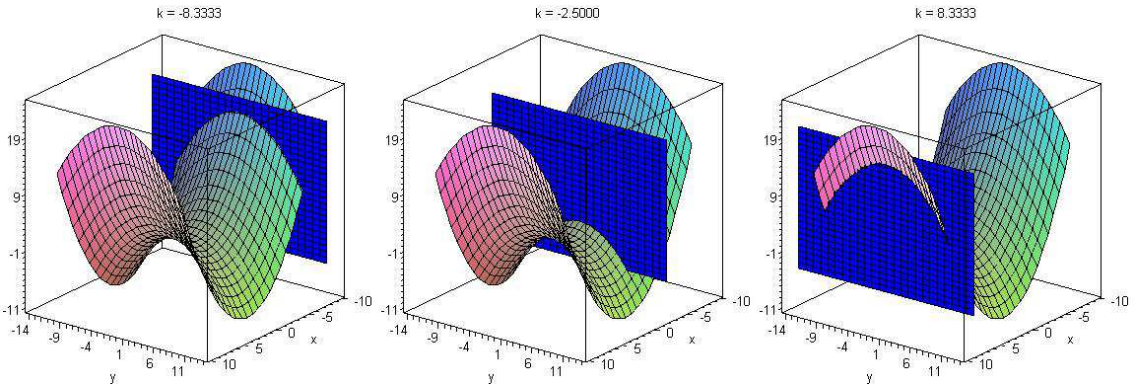
6.5.2. Hiperbolikus paraboloid

```
>hippar:=plot3d(x^2/4-y^2/9,x=-10..10,y=-10..10,axes=box):

>fuggsikok:=animate(plot3d,[[k,y,z],y=-14..14,z=-5..20,color=blue],
```



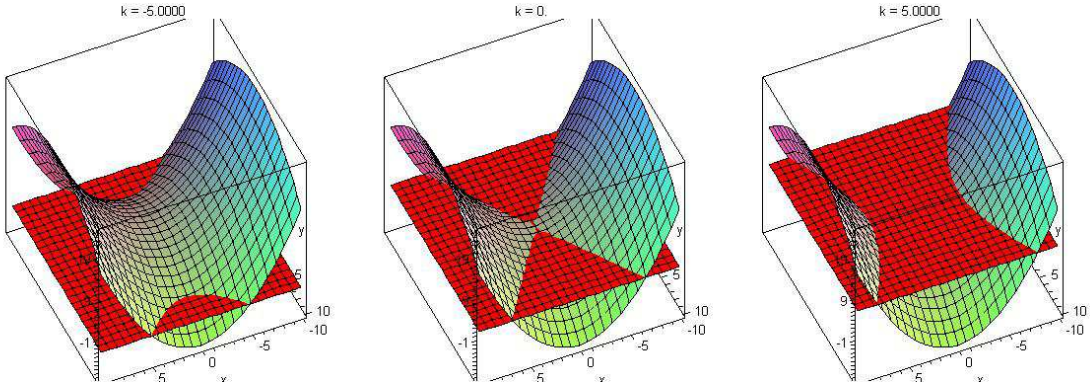
51. ábra. Elliptikus paraboloid vízszintes síkokkal

```
k=-10..10):

>vizsikok:=animate(plot3d,[[x,y,k],x=-10..10,y=-10..10,color=red],
k=-10..20):

>display([hippar,fuggsikok]); (52. ábra)
```



52. ábra. Hiperbolikus paraboloid függőleges síkokkal

```
>display([hippar,vizsikok]); (53. ábra)
```



53. ábra. Hiperbolikus paraboloid vízszintes síkokkal

7. Másodrendű felületek valós síkmetszetei

A másodrendű felületek síkmetszetei szintén másodrendűek (egy sík és egy n-ed rendű felület pontosan n-ed rendű síkgörbében metszi egymást). Ebben a fejezetben megnézzük, hogy az egyes másodrendű felületeknek különböző helyzetű síkokkal milyen metszésvonalai lehetnek.

7.1. Az ellipszoid síkmetszete

Az alcímben azért szerepel síkmetszet (nem pedig síkmetszetei), mert az ellipszoidot bármilyen helyzetű síkkal elmetszve a síkmetszet csak ellipszis lehet (vagy speciális esetben kör). Nézzünk egy példát síkmetszetet szemléltető animáció készítésére:

```
>setoptions3d(scaling=constrained,axes=box):

>a:=5:b:=4:c:=3: # az ellipszis féltengelyei
```

Használjuk az ellipszis paraméteres egyenletét:

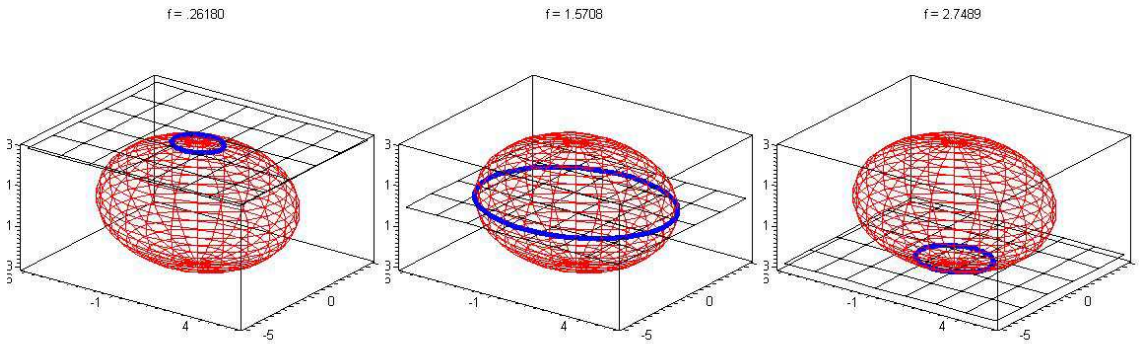
```
>ellipszoid:=plot3d([a*cos(t)*sin(f),b*sin(t)*sin(f),c*cos(f)],
t=0..2*Pi,f=0..Pi,style=line,color=red):
```

Létrehozzuk a metszésvonalat megjelenítő animációt:

```
>gorbe:=animate(spacecurve,[[a*cos(t)*sin(f),b*sin(t)*sin(f),
c*cos(f)],t=0..2*Pi,color=blue,thickness=3],f=0..Pi):
```

Majd a síkot:

```
>sik:=animate(plot3d,[[x,y,c*cos(f)],x=-6..6,y=-5..5,grid=[8,6],
color=black,style=line],f=0..Pi):

>display([ellipszoid,sik,gorbe]); (54. ábra)
```

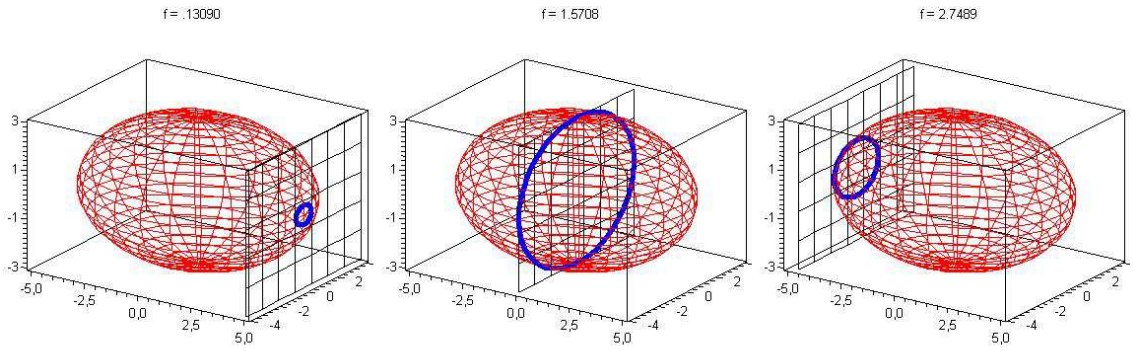


54. ábra.

Most nézzük meg az egyik vízszintes irányban mozgó síkkal a metszésvonalakat:

```
>gorbe:=animate(spacecurve,[[a*cos(f),b*sin(t)*sin(f),c*cos(t)*
sin(f)],t=0..2*Pi,color=blue,thickness=3],f=0..Pi):
```

```
>sik:=animate(plot3d,[[a*cos(f),y,z],y=-4..4,z=-3..3,grid=[8,6],
color=black,style=line],f=0..Pi):

>display([sik,gorbe,ellipszoid]); (55. ábra)
```



55. ábra.

7.2. Az elliptikus kúp síkmetszetei

Az elliptikus kúp esetében már nem olyan egyszerű a helyzet. Itt többféle síkmetszet is lehet a sík helyzetétől függően.

```
>setoptions3d(axes=box):
```

7.2.1. A síkmetszet ellipszis

Ha a metsző sík metszi a kúp összes alkotóját, akkor a síkmetszet ellipszis. Legyen $a = 5$ és $b = 4$, majd térjünk át henger koordináta-rendszerre a kúp kirajzolásához.

```
>a:=5:b:=4:

>kup:=x^2/a^2+y^2/b^2=z^2:

>kup:=subs({x=r*cos(t),y=r*sin(t)},kup):

>collect(kup,r^2):

>kupegy:=sqrt(z^2/(lhs(%)/r^2)):

>kupabra:=plot3d(kupegy,t=0..2*Pi,z=-3..3,coords=cylindrical,
style=line,color=blue):
```

Most kell egy olyan sík, ami a kúp minden alkotóját metszi, legyen ez pl. a $z = \frac{1}{10} \cdot x - k$ egyenletű sík. Az egyenletben a $-k$-val érjük el a sík z tengely menti mozgását majd az animációban. Ezt behelyettesítjük a kúp általános egyenletébe és kifejezzük belőle az y változót. Ezt felhasználva rajzoljuk ki a metszésvonalat.

```
>sik:=animate(plot3d,[1/10*x-k,x=-15..15,y=-10..10,grid=[8,6],
```

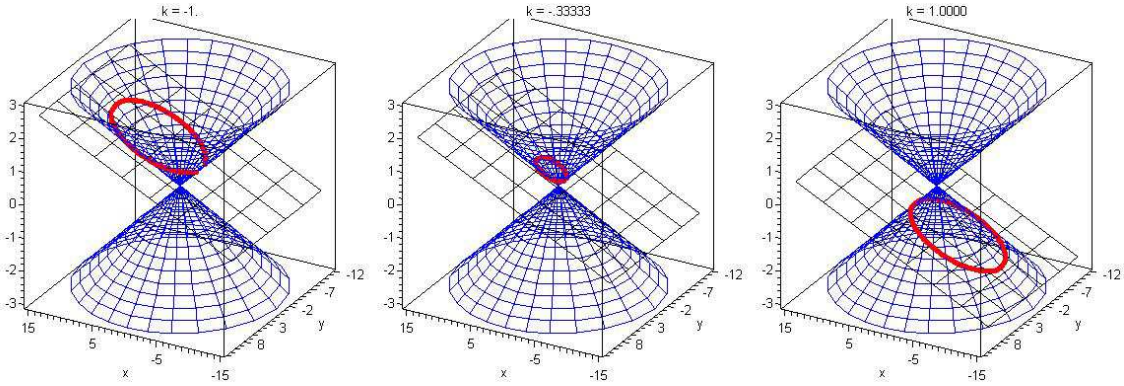
```
style=line,color=black],k=-1..1):

>ellegy:=x^2/a^2+y^2/b^2=(1/10*x-k)^2:

>ym:=solve(ellegy,y):
```

Most következik az ellipszis animációja:

```
>g:=animate(spacecurve,[{[x,ym[1],1/10*x-k],[x,ym[2],1/10*x-k]},
x=-10..10,thickness=3,color=red,numpoints=151],k=-1..1):
```

És a végeredmény (56. ábra):

```
>display([kupabra,sik,g]);
```



56. ábra.

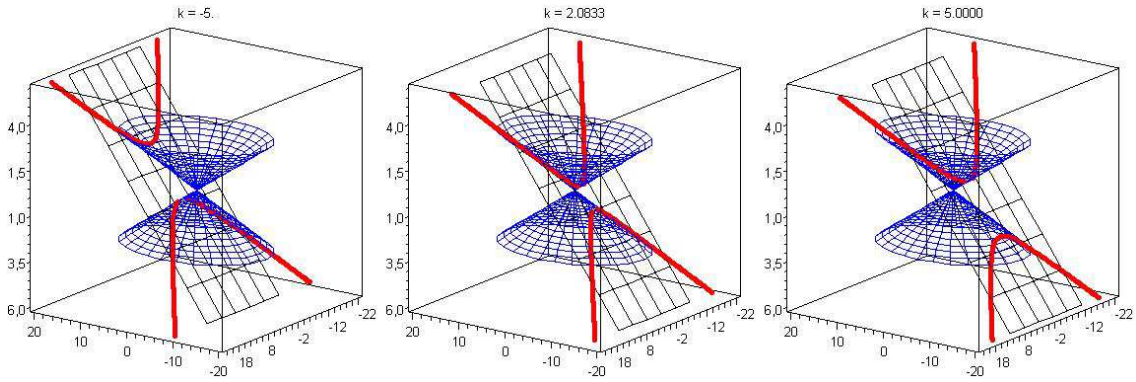
7.2.2. A síkmetszet hiperbola

Ha a metsző sík nem metszi a kúp két alkotóját (két alkotóval párhuzamos) és nem illeszkedik a kúp csúcsára, akkor a síkmetszet hiperbola.
Legyen most a síkunk egyenlete $z = \frac{2}{5} \cdot x$. Az animációban a síkot úgy adjuk meg, hogy az x pontjainak a koordinátái változzanak, így érünk el vízszintes irányú mozgást.

```
>sik:=animate(plot3d,[[x-k,y,2/5*x],x=-15..15,y=-12..12,
grid=[8,6],style=line,color=black],k=-5..5):
```

A kúp egyenletébe behelyettesítjük az x helyére (x-k)-t, a z helyére $\frac{2}{5} \cdot x$-et, majd kifejezzük az y-t:

```
>egy:=(x-k)^2/a^2+y^2/b^2=(2/5*x)^2:

>ym:=solve(egy,y):

>g:=animate(spacecurve,[{[x-k,ym[1],2/5*x],[x-k,ym[2],2/5*x]},
x=-15..15,thickness=3,color=red,numpoints=151],k=-5..5):

>display([kupabra,sik,g]); (57. ábra)
```

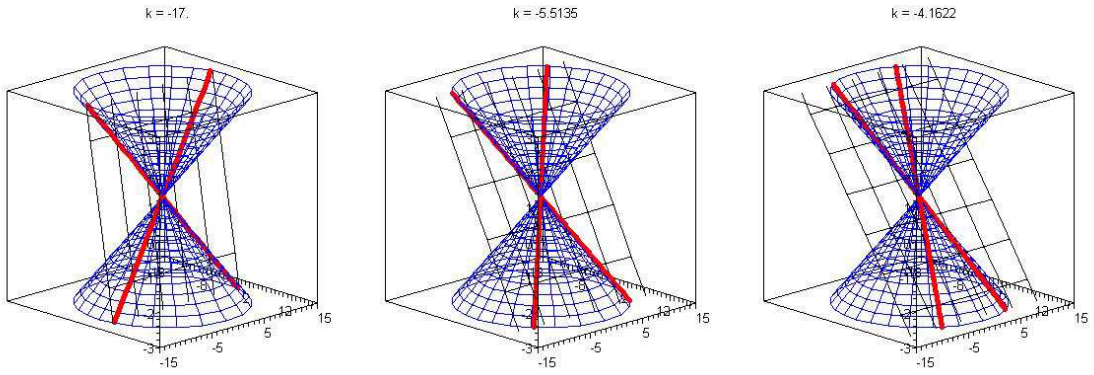


57. ábra.

7.2.3. A síkmetszet egyenespár

Ha a metsző sík illeszkedik a kúp csúcsára, akkor a síkmetszet metsző vagy egybeeső egyenespár (58. ábra).
Az animációban a sík meredekségét változtatva láthatjuk az egyenespárok helyzetét. A síkunk legyen a $z = \frac{k}{18} \cdot x$ egyenlettel megadva:

```
>fszam:=38:m:=18:

>sik:=animate(plot3d,[[x,y,k/m*x],x=-15..15,y=-12..12,
grid=[8,6],style=line,color=black],k=-17..8,frames=fszam):

>egy:=x^2/a^2+y^2/b^2=(k/m*x)^2:

>ym:=solve(egy,y):

>g:=animate(spacecurve,[{[x,ym[1],k/m*x],[x,ym[2],k/m*x]},x=-15..15,
thickness=3,color=red,numpoints=151],k=-17..8,frames=fszam):

>display([kupabra,g,sik],view=[-18..18,-15..15,-3..3]); (58. ábra)
```



58. ábra.

7.2.4. A síkmetszet parabola

Ha a metsző sík párhuzamos a kúp valamely alkotójával, akkor a síkmetszet parabola. Az animációhoz szükségünk van egy olyan síkra, amely egy tetszőleges alkotóval párhuzamos. Célszerű vagy az x, vagy az y tengelyen levő pontokkal számolni. A kúp általános egyenletébe behelyettesítve például az x=3 és y=0 pontokat, a z-t kifejezve kapunk egy számunkra megfelelő síkegyenletet, ez lesz a $z = \frac{1}{5} \cdot x$. A síkunk animációja tehát:

```
>sik:=animate(plot3d,[1/5*x+k,x=-15..15,y=-15..15,style=line,
color=black,grid=[8,6]],k=-5..5):
```

Majd behelyettesítjük a $z = \frac{1}{5} \cdot x + k$ egyenletet a kúp általános egyenletébe, és kifejezzük az x változót:

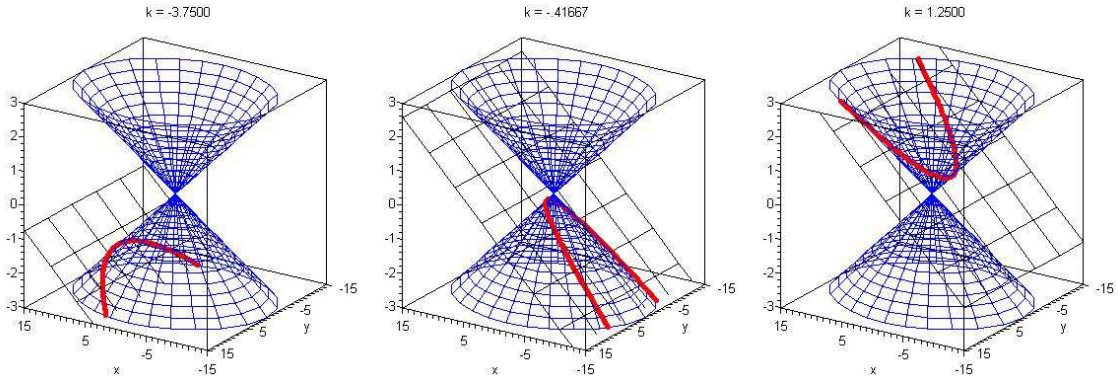
```
>egy:=x^2/a^2+y^2/b^2=(1/5*x+k)^2:

>xm:=solve(egy,x):
```

Most jön a parabolánk animációja:

```
>g:=animate(spacecurve,[[xm,y,1/5*xm+k],y=-15..15,
thickness=3,color=red,numpoints=151],k=-5..5):
```

Majd a végeredmény (59. ábra):

```
>display([kupabra,sik,g],axes=box,view=[-15..15,-15..15,-3..3]);
```



59. ábra.

7.3. Az elliptikus henger síkmetszetei

Az elliptikus henger esetén a metsző sík helyzetétől függően kétféle síkmetszetet kaphatunk.

```
>setoptions3d(axes=box,scaling=constrained):
```

7.3.1. A síkmetszet ellipszis

Ha a metsző sík a henger összes alkotóját metszi, akkor a síkmetszet ellipszis.
Nézzünk egy olyan hengert, amelynél a=5 és b=4, magassága 10. Legyen a síkunk egyenlete $z = \frac{k}{10} \cdot y + 5$. Az animációban a sík meredekségét fogjuk változtatni a k-val.

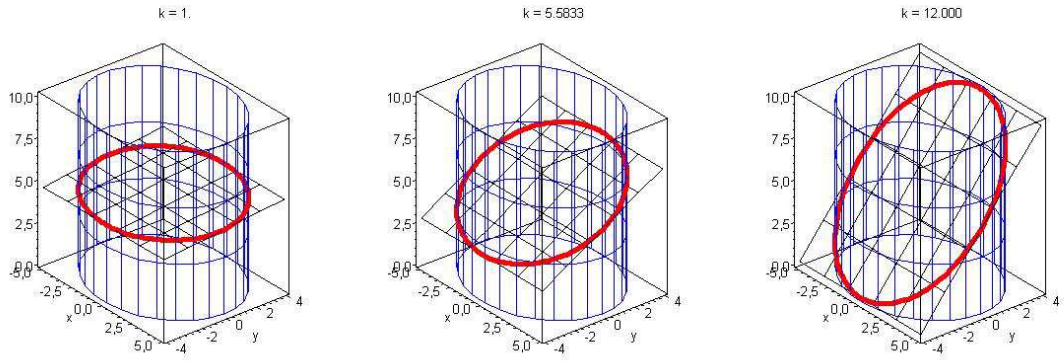
```
>a:=5:b:=4:

>henger:=plot3d([a*cos(u),b*sin(u),v],u=0..2*Pi,v=0..10,style=line,
color=blue,grid=[30,4]):
```

A sík animációja:

```
>sik:=animate(plot3d,[k/10*y+5,x=-5..5,y=-4..4,style=line,
color=black,grid=[6,6]],k=1..12):
```

A síkmetszet pontjait úgy kapjuk meg, hogy a henger egyenletéből kifejezzük az y változót, a görbe z pontjait pedig a sík egyenletébe behelyettesített y koordináták adják.

```
>ykoord:=sqrt(b^2*(1-x^2/a^2)):

>g:=animate(spacecurve,[{[x,ykoord,k/10*ykoord+5],[x,-ykoord,-k/10*
ykoord+5]},x=-5..5,color=red,thickness=3,numpoints=51],k=1..12):

>display([henger,sik,g]); (60. ábra)
```



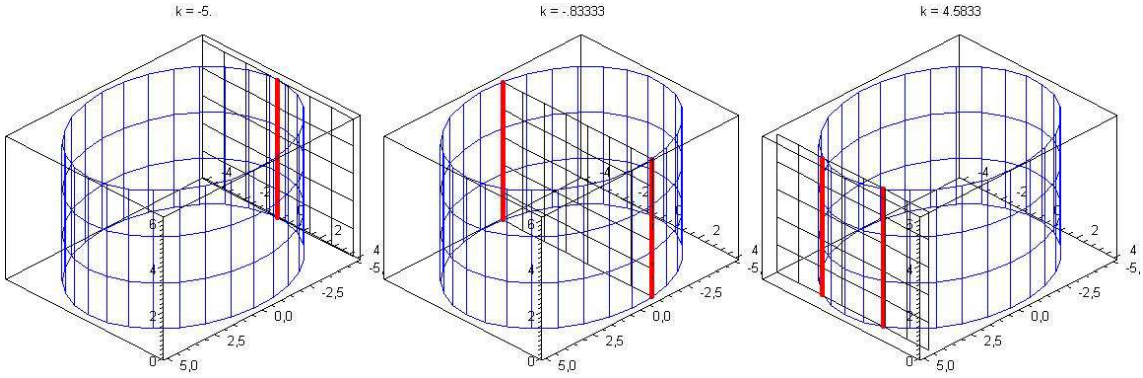
60. ábra.

7.3.2. A síkmetszet egyenespár

Ha a metsző sík párhuzamos az alkotókkal és nem érinti a hengerfelületet, akkor párhuzamos egyenespár, ha érinti azokat, akkor egybeeső egyenespár a síkmetszet.
A síkunkat most az x tengely mentén fogjuk mozgatni és láthatjuk síkmetszetként az egyenespárokat. A sík egyenlete: x=k.

```
>henger:=plot3d([a*cos(u),b*sin(u),v],u=0..2*Pi,v=0..6,style=line,
color=blue,grid=[30,4]):

>sik:=animate(plot3d,[[k,y,z],y=-4..4,z=0..6,grid=[8,6],style=line,
color=black],k=-5..5):

>ykoord:=sqrt(b^2*(1-k^2/a^2)):

>g:=animate(spacecurve,[{[k,ykoord,z],[k,-ykoord,z]},z=0..6,
color=red,thickness=3,numpoints=51],k=-5..5):
```

```
>display([henger,sik,g]); (61. ábra)
```



61. ábra.

7.4. Az elliptikus paraboloid síkmetszetei

7.4.1. A síkmetszet parabola

Ha a metsző sík párhuzamos a paraboloid tengelyével, akkor a síkmetszet parabola. Legyen a paraboloidunk $z = \frac{x^2}{9} + \frac{y^2}{16}$ egyenlettel megadva, a síkunk majd az y tengely mentén fog haladni.

```
>setoptions3d(axes=box):

>a:=3:b:=4:fszam:=21: # frame szám
```

A paraboloidunk létrehozása:

```
>paraboloid:=plot3d(x^2/a^2+y^2/b^2,x=-20..20,y=-10..10,view=0..25,
style=line,color=blue):
```

A síkunk animációjának létrehozása:

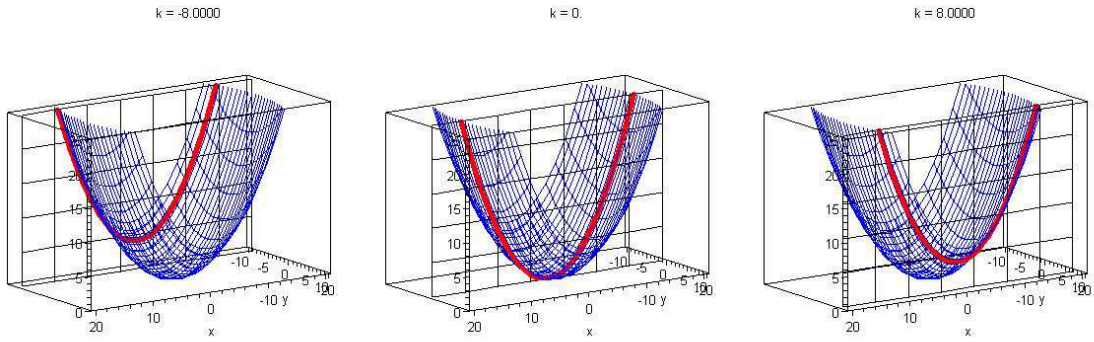
```
>sik:=animate(plot3d,[[x,k,z],x=-20..20,z=0..25,grid=[8,6],
style=line,color=black],k=-10..10,frames=fszam):
```

Mivel a síkot az y tengely mentén mozgatjuk, a paraboloid egyenletéből az x változót kell kifejezni, majd a metszésvonal paraméteres egyenletét a z változó szerint fogjuk felírni:

```
>xkoord:=sqrt(a^2*(z-k^2/b^2)):

>g:=animate(spacecurve,[{[xkoord,k,z],[-xkoord,k,z]},z=0..25,
color=red,thickness=3,numpoints=151],k=-10..10,frames=fszam):

>display([paraboloid,sik,g]); (62. ábra)
```

7.4.2. A síkmetszet ellipszis

Ha a metsző sík nem párhuzamos a forgástengellyel, akkor a síkmetszet ellipszis. Legyen a metsző síkunk a $z = -\frac{1}{4} \cdot x + k$ egyenlettel megadva, ahol k értékét változ-



62. ábra.

tatjuk az animáció során. A paraboloid létrehozása:

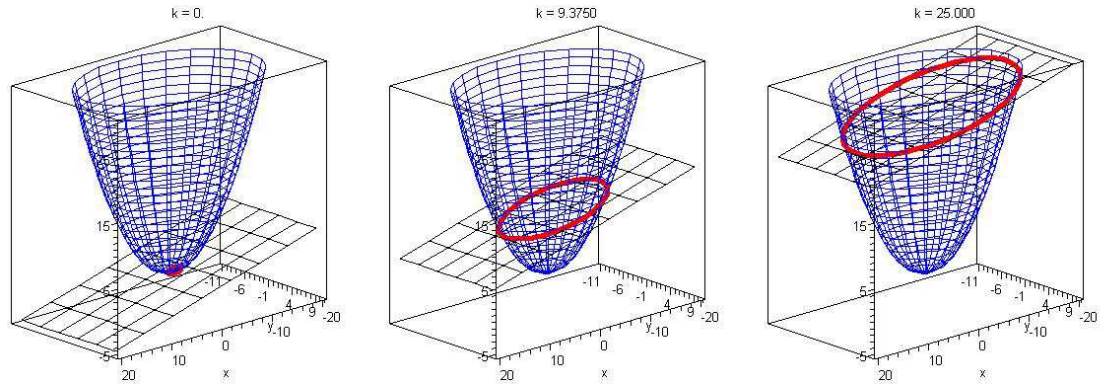
```
>setoptions3d(axes=box,scaling=constrained):

>a:=3:b:=2:

>paraboloid:=plot3d([a*sqrt(u)*cos(v),b*sqrt(u)*sin(v),u],
v=0..2*Pi,u=0..30,style=line,color=blue):
```

A síkunk animációja:

```
>sik:=animate(plot3d,[-1/4*x+k,x=-20..20,y=-10..10,grid=[8,6],
style=line,color=black],k=0..25):
```

A metszésvonal paraméteres egyenletét úgy kapjuk meg, hogy a sík általános egyenletébe behelyettesítjük a paraboloid paraméteres egyenletét és ebből kifejezzük u-t.

```
>egy:=u=-1/4*(a*sqrt(u)*cos(v))+k:

>um:=solve(egy,u):

>g:=animate(spacecurve,[[a*sqrt(um[1])*cos(v),b*sqrt(um[1])*sin(v),
um[1]],v=0..2*Pi,color=red,thickness=3],k=0..25):

>display([paraboloid,sik,g]); (63. ábra)
```



63. ábra.

7.5. A hiperbolikus henger síkmetszetei

```
>setoptions3d(axes=box):
```

7.5.1. A síkmetszet egyenespár

Ha a sík párhuzamos az alkotókkal vagy érinti a hengert, akkor a síkmetszet egy párhuzamos vagy egybeeső egyenespár.
Legyen a=3 és b=2. A hiperbolikus henger paraméteres egyenletét használjuk.

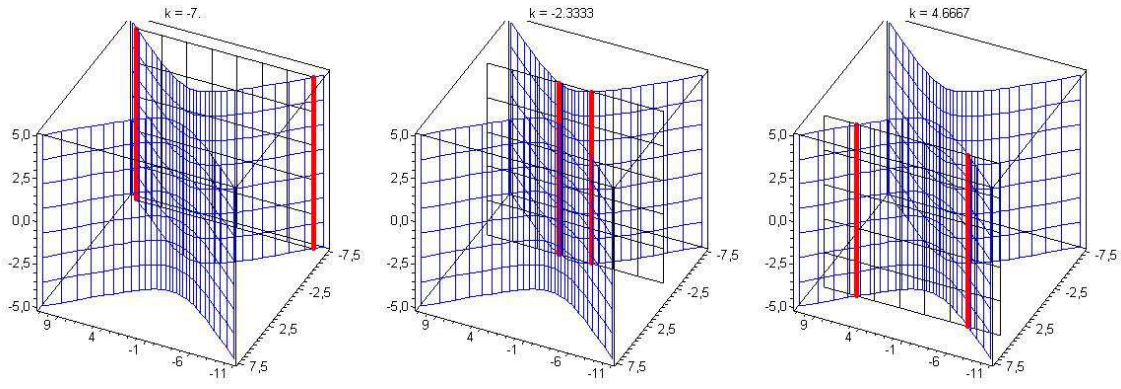
```
>a:=3:b:=2:

>henger:=plot3d({[a*sinh(u),b*cosh(u),v],[-a*sinh(u),-b*cosh(u),v]},
u=-2..2,v=-5..5,style=line,color=blue,grid=[30,8]):
```

A sík animációja:

```
>sik:=animate(plot3d,[[x,k,z],x=-10..10,z=-5..5,grid=[8,6],
style=line,color=black],k=-7..7):
```

A hiperbolikus henger általános egyenletébe behelyettesítjük a sík egyenletét és kifejezzük az x változót:

```
>xm:=sqrt(a^2*(k^2/b^2-1)):

>g:=animate(spacecurve,[{[xm,k,z],[-xm,k,z]},z=-5..5,color=red,
thickness=3],k=-7..7):

>display([henger,sik,g]); (64. ábra)
```



64. ábra.

7.5.2. A síkmetszet hiperbola

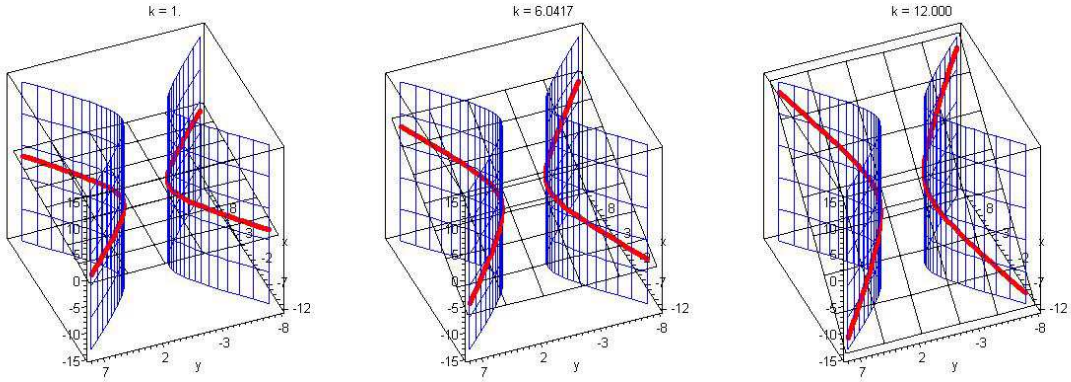
Ha a sík nem párhuzamos az alkotókkal, akkor a síkmetszet hiperbola.
Legyen most a síkunk egyenlete $z = \frac{k}{10} \cdot x$, ahol a k értékét változtatjuk, így a különböző meredekségű síkokkal való síkmetszeteket láthatjuk:

```
>henger:=plot3d({[a*sinh(u),b*cosh(u),v],[-a*sinh(u),-b*cosh(u),v]},
u=-2..2,v=-15..15,style=line,color=blue,grid=[30,6]):
```

```
>sik:=animate(plot3d,[k/10*x,x=-12..12,y=-8..8,grid=[8,6],
style=line,color=black],k=1..12):
```

A síkmetszet paraméteres egyenletét megkapjuk, ha a sík egyenletébe behelyettesítjük a henger paraméteres egyenletét, majd kifejezve a v-t és visszaírva a henger egyenletébe megkapjuk a görbeegyenletet:

```
>g:=animate(spacecurve,[{[a*sinh(u),b*cosh(u),k/10*a*sinh(u)],[-a*
sinh(u),-b*cosh(u),-k/10*a*sinh(u)]},u=-2..2,color=red,
thickness=3],k=1..12):

>display([henger,sik,g]); (65. ábra)
```



65. ábra.

7.6. A hiperbolikus paraboloid síkmetszetei

Ezt a felületet szokták nyeregfelületnek is nevezni, melynek a síkok helyzetétől függően háromféle síkmetszete lehet:

1. Amennyiben a sík a nyeregfelület nyeregpontján ül, akkor a síkmetszet egy metsző egyenespár.
2. Amennyiben a sík párhuzamos a nyereg lapnormálisával, akkor a síkmetszet parabola.
3. Minden más esetben a síkmetszet hiperbola.

Az 1. és 3. esetet együtt fogjuk szemléltetni.

```
>setoptions3d(axes=box):
```

7.6.1. A síkmetszet egyenespár vagy hiperbola

`>a:=5:b:=4:` A hiperbolikus paraboloid z=xy alakú egyenletét használjuk az ábra kirajzolásához és a metszésvonal meghatározásához is.

```
>hippar:=plot3d(x*y,x=-50..50,y=-50..50,style=line,color=blue):
```

A síkunk egyenlete z=k, ahol k fog változni:

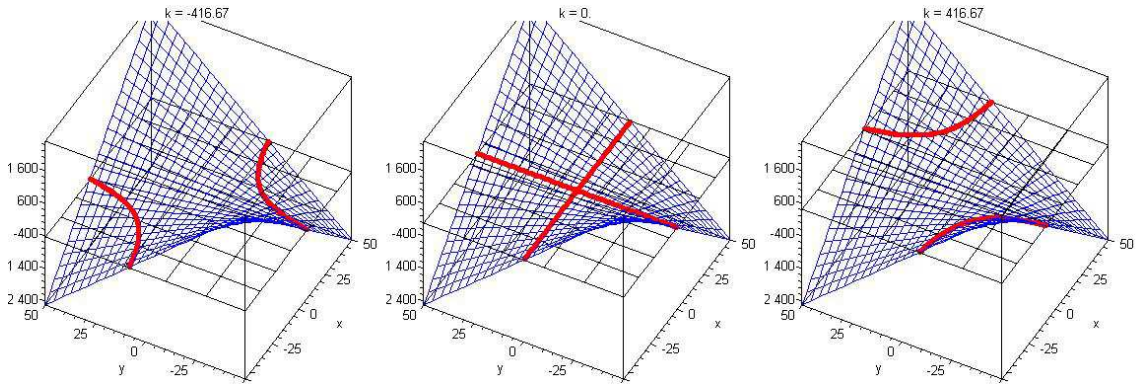
```
>sik:=animate(plot3d,[[x,y,k],x=-50..50,y=-50..50,grid=[8,6],
style=line,color=black],k=-2500..2500):
```

A görbéket 4 animációval jelenítjük meg:

```
>g1a:=animate(spacecurve,[[x,k/x,k],x=-50..-0.0001,color=red,
thickness=3],k=-2500..2500):

>g1b:=animate(spacecurve,[[x,k/x,k],x=0.0001..50,color=red,
thickness=3],k=-2500..2500):

>g2a:=animate(spacecurve,[[k/y,y,k],y=-50..-0.0001,color=red,
thickness=3],k=-2500..2500):

>g2b:=animate(spacecurve,[[k/y,y,k],y=0.0001..50,color=red,
thickness=3],k=-2500..2500):
```

A `g1` és `g2` görbéknek azért van `a` és `b` része, mert $\frac{k}{x}$-nek és $\frac{k}{y}$-nak 0-ban szakadása van, így megakadályozzuk egy x=0 és y=0 egyenes megjelenését az animációban. Azért van külön `g1` és `g2` görbepár, mert k=0 esetén így láthatjuk a metsző egyenespárt (egyébként a `g1a=g2b` és `g1b=g2a`).

```
>display([hippar,sik,g1a,g2b,g2a,g2b],view=[-50..50,-50..50,
-2500..2500]); (66. ábra)
```



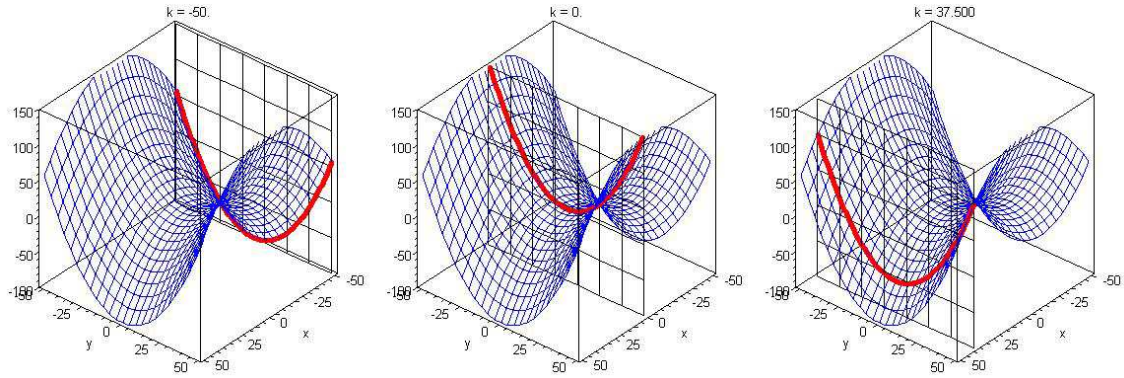
66. ábra.

7.6.2. A síkmetszet parabola

A parabola síkmetszethez a hiperbolikus paraboloid másik felírási módját választjuk a szemléletesebb megjelenítéshez:

```
>hippar:=plot3d(y^2/b^2-x^2/a^2,x=-50..50,y=-50..50,style=line,
color=blue):

>sik:=animate(plot3d,[[k,y,z],y=-50..50,z=-100..150,grid=[8,6],
style=line,color=black],k=-50..50):

>zm:=y^2/b^2-k^2/a^2:
```

```
>g:=animate(spacecurve,[{[k,y,zm],[k,-y,zm]},y=-50..50,color=red,
thickness=3,numpoints=151],k=-50..50):

>display([hippar,sik,g],view=-100..150); (67. ábra)
```



67. ábra.

7.7. Az egyköpenyű hiperboloid síkmetszetei

Az egyköpenyű hiperboloid síkmetszeteinek tanulmányozásához szükséges egy új fogalom bevezetése. Tudjuk azt, hogy a hiperbola képzetes tengelye körüli megforgatásával egyköpenyű hiperboloidot kapunk. A hiperbola aszimptotáit megforgatva (szintén a képzetes tengely körül) kúpfelületet kapunk, ezt nevezik a hiperboloid aszimptotikus kúpjának.

7.7.1. A síkmetszet ellipszis

Amennyiben a sík forgástengellyel bezárt szöge nagyobb, mint az aszimptotikus kúp félnyílásszöge, akkor a síkmetszet ellipszis.
Legyen a=2 és c=2, használjuk a hiperboloid paraméteres egyenletét:

```
>a:=2:c:=2:

>egykop:=plot3d([a*sqrt(1+u^2)*cos(v),a*sqrt(1+u^2)*sin(v),c*u],
v=0..2*Pi,u=-c..c,style=line,color=blue):
```

A síkunk egyenlete legyen $z = \frac{2}{5} \cdot x + k$, ahol k-t fogjuk változtatni, a sík a z tengely mentén fog mozogni.

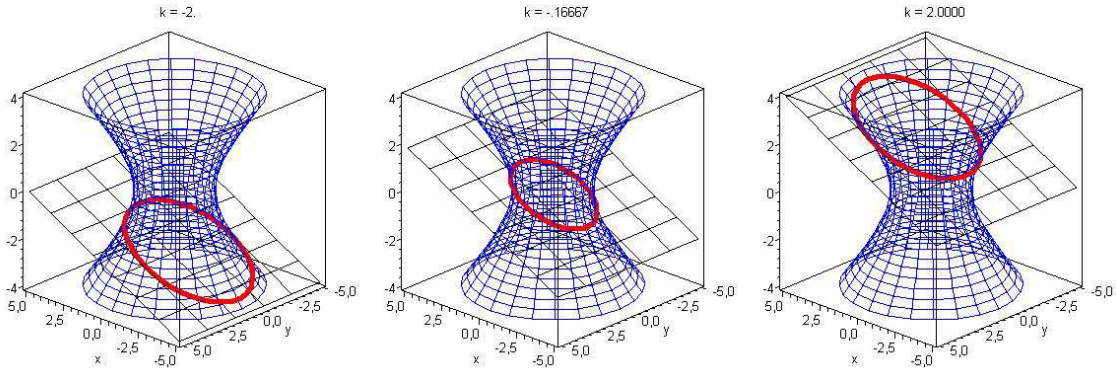
```
>sik:=animate(plot3d,[2/5*x+k,x=-5..5,y=-5..5,style=line,grid=[8,6],
color=black],k=-2..2):
```

A síkmetszethez a hiperboloid paraméteres egyenletét behelyettesítjük a sík egyenletébe és kifejezzük u-t.

```
>egyenlet:=c*u=2/5*(a*sqrt(1+u^2)*cos(v))+k:

>um:=solve(egyenlet,u):
```

Két megoldást kapunk a négyzetre emelés miatt, ezek közül a síkunkhoz csak az egyik jó.

```
>g:=animate(spacecurve,[[a*sqrt(1+um[2]^2)*cos(v),a*sqrt(1+um[2]^2)*
sin(v),c*um[2]],v=0..2*Pi,color=red,thickness=3],k=-2..2):

>display([egykop,sik,g]); (68. ábra)
```



68. ábra.

7.7.2. A síkmetszet parabola

Amennyiben a sík forgástengellyel bezárt szöge megegyezik az aszimptotikus kúp félnyílásszögével, akkor a síkmetszet parabola.
Ha a sík érinti az aszimptotikus kúpot, akkor a hiperboloid síkmetszete egy párhuzamos egyenespár.
Ez az animáció érdekes, hiszen itt lesz a síkmetszet parabola és párhuzamos egyenespár. A hiperboloidunk nem változik, ugyanaz mint az előbbi részben.
A síkunk animációja:

```
>sik:=animate(plot3d,[x+k,x=-5..5,y=-5..5,style=line,grid=[8,6],
color=black],k=-3..2):
```

Szintén behelyettesítéssel és az u paraméter kifejezésével kapjuk meg síkmetszetünket (az u értékére itt is csak az egyik megoldás jó, mint az előbb).

```
>egyenlet:=c*u=a*sqrt(1+u^2)*cos(v)+k:

>um:=solve(egyenlet,u):
```

A síkmetszet animációja:

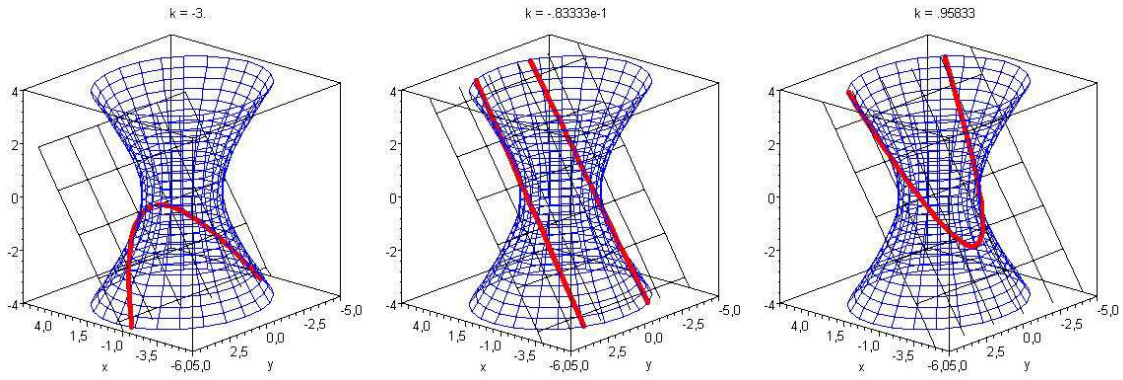
```
>g:=animate(spacecurve,[[a*sqrt(1+um[2]^2)*cos(v),a*sqrt(1+um[2]^2)*
sin(v),c*um[2]],v=0.0001..2*Pi,color=red,thickness=3],k=-3..2):
```

Mivel az `um[2]` megoldásban a nevező nem értelmezhető 0-nál, fontos hogy a `v`-t ne 0-tól kezdjük, különben az animációban hibát fogunk tapasztalni.

```
>display([egykop,sik,g],view=[-6..6,-5..5,-4..4]); (69. ábra)
```

7.7.3. A síkmetszet hiperbola

Amennyiben a sík forgástengellyel bezárt szöge kisebb, mint az aszimptotikus kúp félnyílásszöge, akkor a síkmetszet hiperbola vagy speciális esetben metsző egyenespár.



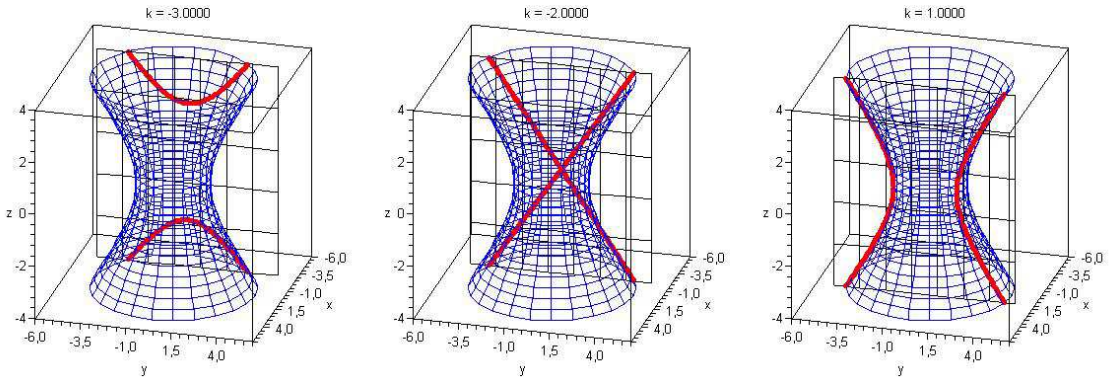
69. ábra.

A sík animációja:

```
>sik:=animate(plot3d,[[k,y,z],y=-5..5,z=-4..4,labels=["x","y","z"],
style=line,grid=[8,6],color=black],k=-4..4):
```

A hiperboloid általános egyenletébe helyettesítjük be a sík paraméteres egyenletét:

```
>egyenlet:=k^2/a^2+y^2/a^2-z^2/c^2=1:

>um:=solve(egyenlet,z):

>g:=animate(spacecurve,[{[k,y,um[1]],[k,-y,um[2]},y=-5..5,color=red,
thickness=3,numpoints=151],k=-4..4):

>display([egykop,sik,g],view=[-6..6,-6..6,-4..4]); (70. ábra)
```



70. ábra.

7.8. A kétköpenyű hiperboloid síkmetszetei

Ugyanúgy, mint az egyköpenyű hiperboloid esetén, itt is beszélhetünk aszimptotikus kúpról.

7.8.1. A síkmetszet ellipszis

Amennyiben a sík forgástengellyel bezárt szöge nagyobb az aszimptotikus kúp félnyílásszögénél, akkor a síkmetszet ellipszis.
Legyen a=2 és c=1, használjuk a paraméteres egyenletet.

```
>a:=2:c:=1:

>ketkop:=plot3d({[a*sinh(u)*cos(v),a*sinh(u)*sin(v),c*cosh(u)],
[a*sinh(u)*cos(v),a*sinh(u)*sin(v),-c*cosh(u)]},u=-2..2,v=0..Pi,
style=line,color=blue):
```

A síkunk legyen a $z = \frac{1}{6} \cdot x + k$ egyenletű:

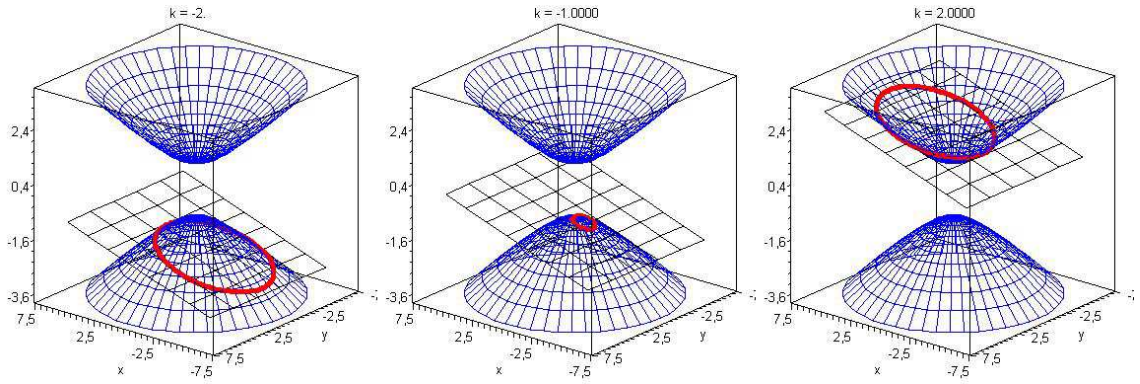
```
>sik:=animate(plot3d,[1/6*x+k,x=-6..6,y=-6..6,style=line,grid=[8,6],
color=black],k=-2..2):
```

A sík egyenletébe behelyettesítjük a hiperboloid paraméteres egyenletét, majd kifejezzük u-t.

```
>egyenlet:=-c*cosh(u)=1/6*a*sinh(u)*cos(v)+k:

>um:=solve(egyenlet,u):
```

A kapott gyököket visszahelyettesítve a hiperboloid paraméteres egyenletébe megkapjuk az ellipszis síkmetszetét.

```
>g:=animate(spacecurve,[{[a*sinh(um[1])*cos(v),a*sinh(um[1])*sin(v),
-c*cosh(um[1])],[a*sinh(um[2])*cos(v),a*sinh(um[2])*sin(v),-c*
cosh(um[2])]},v=0..Pi,color=red,thickness=3],k=-2..2):

>display([ketkop,sik,g]); (71. ábra)
```



71. ábra.

7.8.2. A síkmetszet parabola

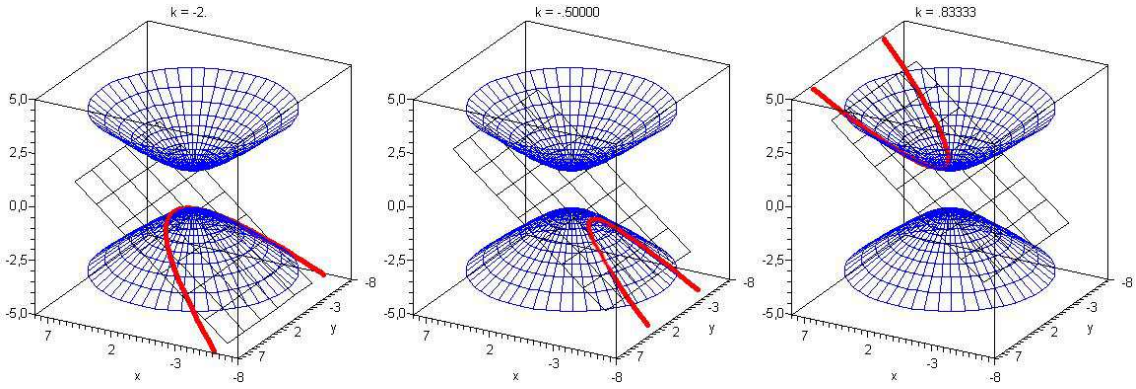
Ha a sík forgástengellyel bezárt szöge megegyezik az aszimptotikus kúp félnyílásszögével, akkor a síkmetszet parabola.
A síkunk egyenlete $z = \frac{1}{2} \cdot x + k$:

```
>sik:=animate(plot3d,[1/2*x+k,x=-6..6,y=-6..6,style=line,grid=[8,6],
color=black],k=-2..2):
```

```
>egyenlet:=-c*cosh(u)=1/2*a*sinh(u)*cos(v)+k:

>um:=solve(egyenlet,u):
```

Mivel a kapott gyököknek Π-ben szakadásuk van, ezért a görbék kirajzolásánál a Π-nél nyílt lesz az intervallum.

```
>g:=animate(spacecurve,[{[a*sinh(um[1])*cos(v),a*sinh(um[1])*sin(v),
-c*cosh(um[1])],[a*sinh(um[2])*cos(v),a*sinh(um[2])*sin(v),-c*
cosh(um[2])]},v=0..Pi-0.0001,color=red,thickness=3,numpoints=151],
k=-2..2):

>display([ketkop,sik,g]); (72. ábra)
```



72. ábra.

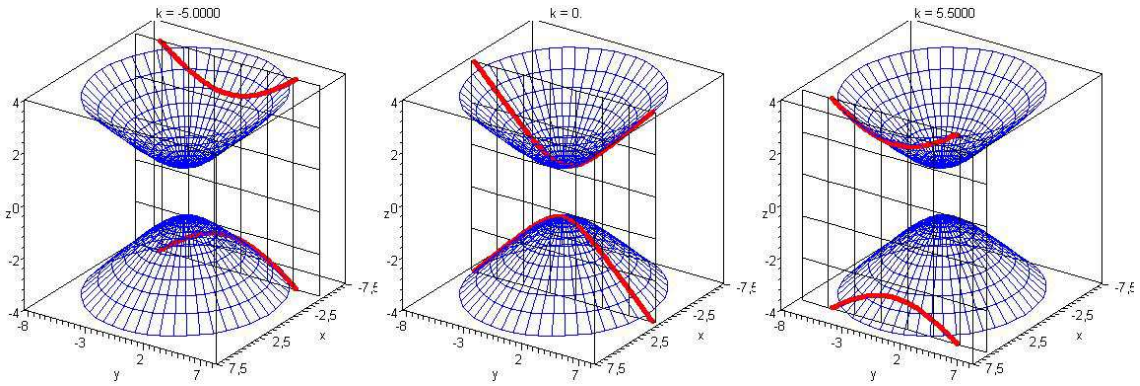
7.8.3. A síkmetszet hiperbola

Ha a sík forgástengellyel bezárt szöge kisebb az aszimptotikus kúp félnyílásszögénél, akkor a síkmetszet hiperbola.

```
>sik:=animate(plot3d,[[k,y,z],y=-8..8,z=-4..4,style=line,grid=[8,6],
color=black,labels=["x","y","z"]],k=-6..6):
```

A síkmetszet egyenletéhez a hiperboloid általános egyenletébe helyettesítjük a sík paraméteres egyenletét és kifejezzük a z változót.

```
>egyenlet:=k^2/a^2+y^2/a^2-z^2/c^2=-1:

>zm:=solve(egyenlet,z):

>g:=animate(spacecurve,[{[k,y,zm[1]],[k,y,zm[2]]},y=-8..8,color=red,
thickness=3,numpoints=151],k=-6..6):

>display([ketkop,sik,g],view=-4..4); (73. ábra)
```



73. ábra.

7.9. A parabolikus henger síkmetszetei

7.9.1. A síkmetszet egyenespár

Abban az esetben, ha a sík normálvektora merőleges a henger alkotóira, akkor a síkmetszet egy párhuzamos egyenespár (vagy egybeeső, ha a sík érinti a felületet). Legyen r=2 a parabolikus hengernél.

```
>r:=2:

>henger:=plot3d(-x^2/(2*r),x=-5..5,y=-5..5,style=line,color=blue):
```

A síkunk z=k egyenletű:

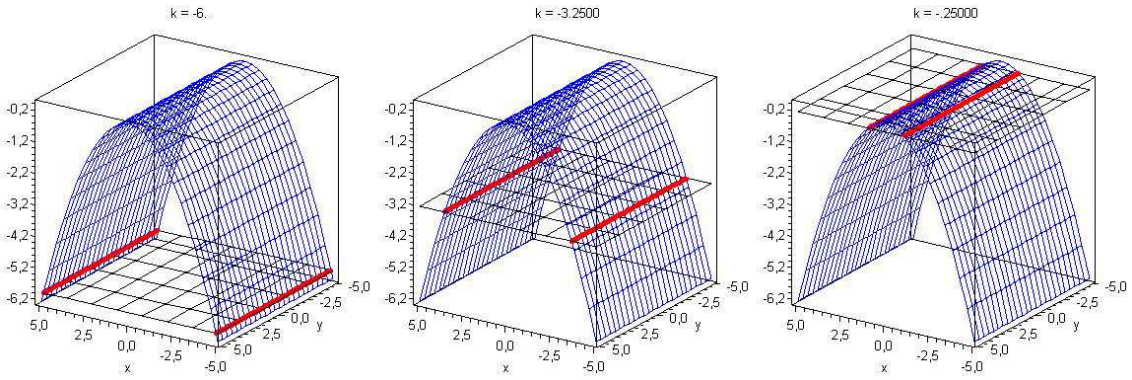
```
>sik:=animate(plot3d,[[x,y,k],x=-5..5,y=-5..5,grid=[8,6],
style=line,color=black],k=-6..0):

>egyenlet:=x^2+2*r*k=0:

>xm:=solve(egyenlet,x):

>g:=animate(spacecurve,[{[xm[1],y,k],[xm[2],y,k]},y=-5..5,
style=line,color=red,thickness=3],k=-6..0):

>display([henger,sik,g]); (74. ábra)
```

7.9.2. A síkmetszet parabola

Amennyiben a sík normálvektora nem merőleges a henger alkotóira, akkor a síkmetszet parabola.

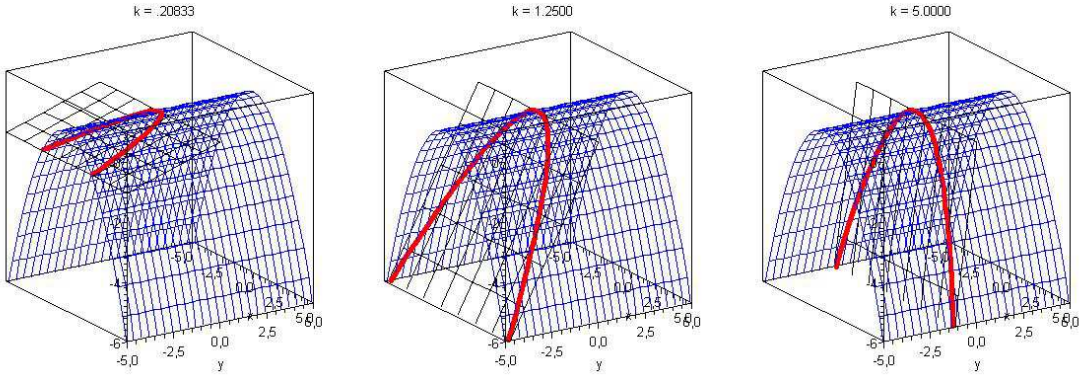
A sík legyen most z=yk egyenlettel megadva.

```
>sik:=animate(plot3d,[y*k,x=-5..5,y=-5..0,grid=[8,6],style=line,
color=black],k=0..5):

>egyenlet:=x^2+2*r*y*k=0:

>ym:=solve(egyenlet,y):
```



74. ábra.

```
>g:=animate(spacecurve,[[x,ym,ym*k],x=-5..5,style=line,color=red,
thickness=3],k=0..5):

>display([henger,sik,g],view=[-5..5,-5..5,-6..1]); (75. ábra)
```



75. ábra.

8. Áthatások

A mérnöki és műszaki területeken fontos szerepe van a testek és felületek áthatásának. Az áthatás alatt azt értjük, hogy a testek illetve a felületek egymáshoz csatlakozva milyen alakzatokban (görbékben) metszik egymást.
Mivel a szakdolgozat célja a felületek (azon belül a másodrendű felületek) megjelenítése és vizsgálata Maple-ben, ezért a másodrendű felületek áthatásával, a felületek áthatási görbéjének szemléltetésével fogunk megismerkedni ebben a részben.
Tudjuk azt, hogy egy n-ed rendű illetve egy k-ad rendű algebrai felület egymást egy n·k-ad rendű áthatási görbében metszi. A mi esetünkben az áthatási görbék 4-ed rendűek lesznek.

8.1. Az ellipszoid áthatási görbéi

8.1.1. Az ellipszoid áthatása elliptikus hengerrel

Elsőként az ellipszoid áthatását egy elliptikus hengerrel fogjuk megnézni.
Magánál a 2 felület ábrázolásánál a paraméteres egyenleteket használjuk. Az animációban a henger az x tengely mentén mozog, és nézhetjük az ellipszoiddal való áthatását.
Az áthatási görbét úgy tudjuk meghatározni, hogy az ellipszoid általános egyenletébe behelyettesítjük a henger paraméteres egyenletét, majd ebből kifejezzük az egyik paraméter (esetünkben a v paraméter) értékét. Ezt visszaírva a henger paraméteres egyenletébe megkapjuk a görbe paraméteres egyenletét.
Nézzük a megoldást:

```
>a:=4:b:=3:c:=2:d:=2:e:=2:

>ellabra:=plot3d([a*cos(t)*sin(f),b*sin(t)*sin(f),c*cos(f)],
t=0..2*Pi,f=0..Pi,axes=box,scaling=constrained,style=line,
color=blue,grid=[15,15]):

>hengerabra:=animate(plot3d,[[d*cos(u)+k,e*sin(u),v],u=0..2*Pi,
v=-3..3,style=line,color=green,grid=[15,2]],k=-7..7):

>ell:=x^2/a^2+y^2/b^2+z^2/c^2=1:
```

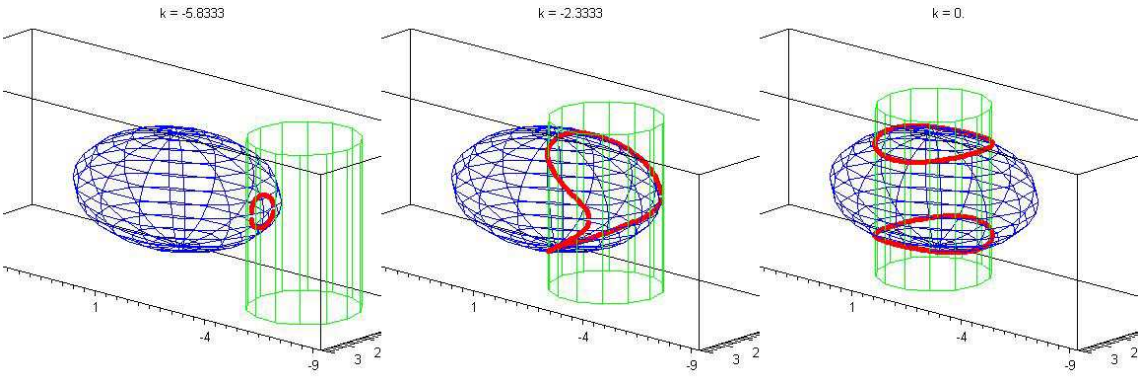
Most behelyettesítjük a henger paraméteres egyenletét:

```
>ellhely:=subs({x=d*cos(u)+k,y=e*sin(u),z=v},ell):

>vm:=solve(ellhely,v):
```

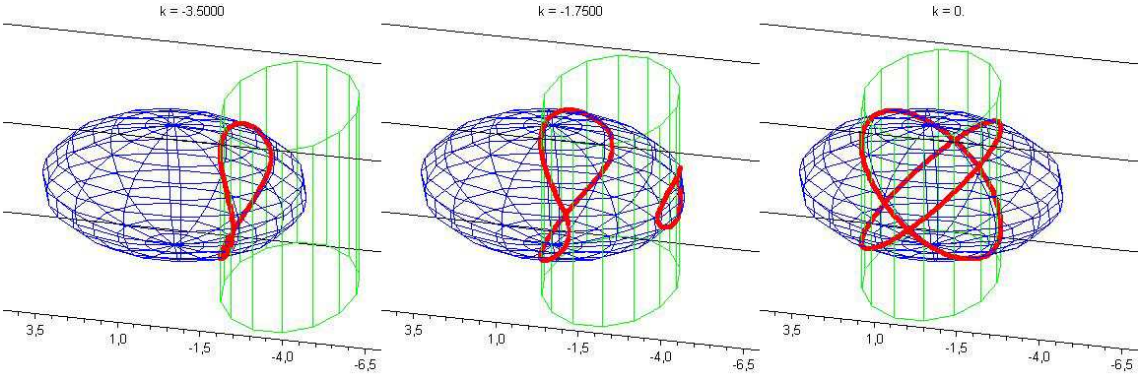
Most jönnek az áthatási görbék animációi:

```
>ag:=animate(spacecurve,[{[d*cos(u)+k,e*sin(u),vm[1]],[d*cos(u)+k,
e*sin(u),vm[2]]},u=0..2*Pi,color=red,thickness=3,numpoints=151],
k=-7..7):

>display([ellabra,hengerabra,ag]); (76. ábra)
```

Érdemes megnézni az `e=3` esetet is: (77. ábra)



76. ábra.



77. ábra.

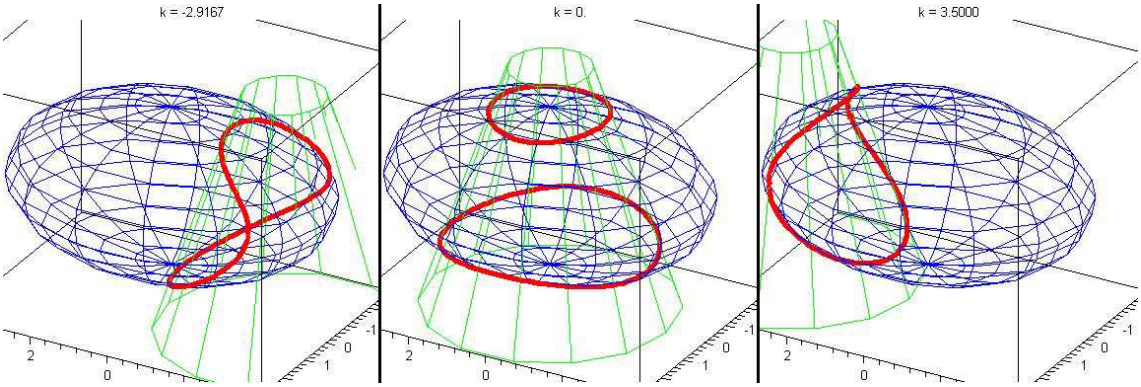
8.1.2. Ellipszoid áthatása elliptikus kúppal

Itt is hasonlóan fogunk eljárni mint az előbb, a kúpot mozgatjuk az x tengely mentén, az ellipszoid általános egyenletébe helyettesítjük a kúp paraméteres egyenletét és az u paramétert kifejezzük.

```
>a:=4:b:=3:c:=2:

>d:=2:e:=2:h:=6:

>ellabra:=plot3d([a*cos(t)*sin(f),b*sin(t)*sin(f),c*cos(f)],
t=0..2*Pi,f=0..Pi,axes=box,scaling=constrained,style=line,
color=blue,grid=[15,15]):

>kupabra:=animate(plot3d,[[d*(h-u)/h*cos(v)+k,e*(h-u)/h*sin(v),u],
v=0..2*Pi,u=0-3..h-3,style=line,color=green,grid=[15,2]],k=-7..7):

>ell:=x^2/a^2+y^2/b^2+z^2/c^2=1:

>ellhely:=subs({x=d*(h-u)/h*cos(v)+k,y=e*(h-u)/h*sin(v),z=u},ell):
>um:=solve(ellhely,u):

>ag:=animate(spacecurve,[{[d*(h-um[1])/h*cos(v)+k,e*(h-um[1])/h*
```

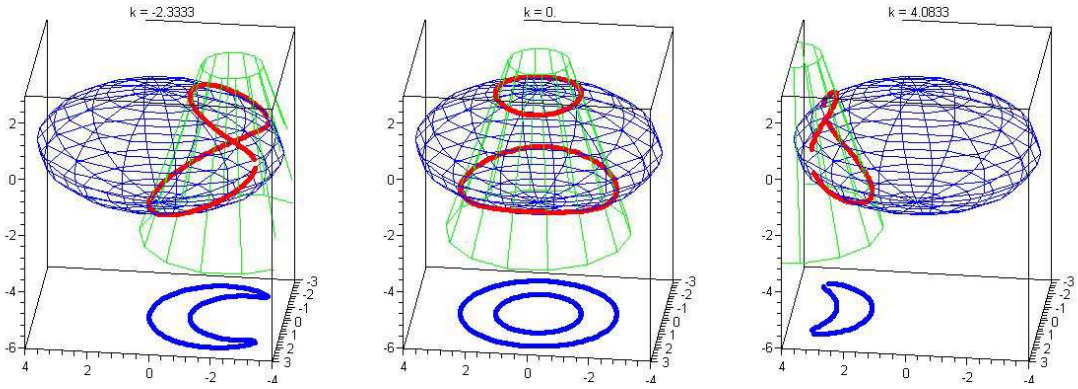
```
sin(v),um[1]],[d*(h-um[2])/h*cos(v)+k,e*(h-um[2])/h*sin(v),um[2]]},
v=0..2*Pi,color=red,thickness=3,numpoints=151],k=-7..7):

>display([ellabra,kupabra,ag],view=[-4..4,-3..3,-3..3]); (78. ábra)
```

Érdekességképpen még kirajzoltathatjuk az áthatási görbék függőleges vetületeit:



78. ábra.

```
>agv:=animate(spacecurve,[{[d*(h-um[1])/h*cos(v)+k,e*(h-um[1])/h*
sin(v),-6],[d*(h-um[2])/h*cos(v)+k,e*(h-um[2])/h*sin(v),-6]},
v=0..2*Pi,color=blue,thickness=3,numpoints=151],k=-7..7):

>display([ellabra,kupabra,ag,agv],view=[-4..4,-3..3,-6..3]);
(79. ábra)
```



79. ábra.

8.2. Az elliptikus kúp áthatási görbéi

8.2.1. Elliptikus kúp áthatása elliptikus hengerrel

```
>a:=2:b:=2:h:=4:
>d:=1:e:=1:f:=4:

>kupabra:=plot3d([a*(h-u)/h*cos(v),b*(h-u)/h*sin(v),u],v=0..2*Pi,
```

```
u=0..h,axes=box,style=line,color=blue,grid=[15,4]):

>hengerabra:=animate(plot3d,[[d*cos(u)+k,e*sin(u),v],u=0..2*Pi,
v=0..f,grid=[15,2],style=line,color=green],k=-4..4):
```

A henger általános egyenletébe behelyettesítjük a kúp paraméteres egyenletét, majd kifejezzük u-t.

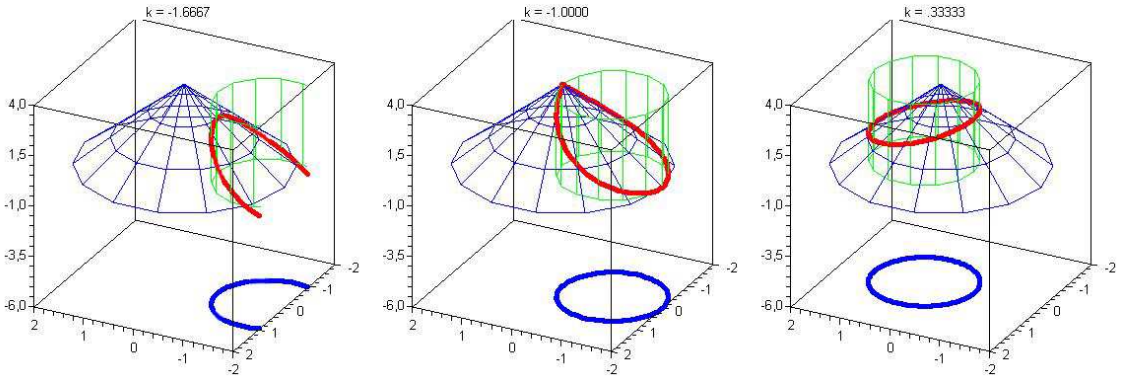
```
>hengeregy:=(x-k)^2/d^2+y^2/e^2=1:

>hengeregyhely:=subs({x=a*(h-u)/h*cos(v),y=b*(h-u)/h*sin(v),z=u},
hengeregy):

>um:=solve(hengeregyhely,u):
```

A kapott értékeket a kúp paraméteres egyenletébe helyettesítve megkapjuk az áthatási görbét:

```
>ag:=animate(spacecurve,[{[a*(h-um[1])/h*cos(v),b*(h-um[1])/h*
sin(v),um[1]],[a*(h-um[2])/h*cos(v),b*(h-um[2])/h*sin(v),um[2]]},
v=0..2*Pi,color=red,thickness=3,numpoints=151],k=-4..4):
```

A görbék függőleges vetületei:

```
>agv:=animate(spacecurve,[{[a*(h-um[1])/h*cos(v),b*(h-um[1])/h*
sin(v),-6],[a*(h-um[2])/h*cos(v),b*(h-um[2])/h*sin(v),-6]},
v=0..2*Pi,color=blue,thickness=3,numpoints=151],k=-4..4):

>display([kupabra,hengerabra,ag,agv],view=[-2..2,-2..2,-6..4]);
(80. ábra)
```

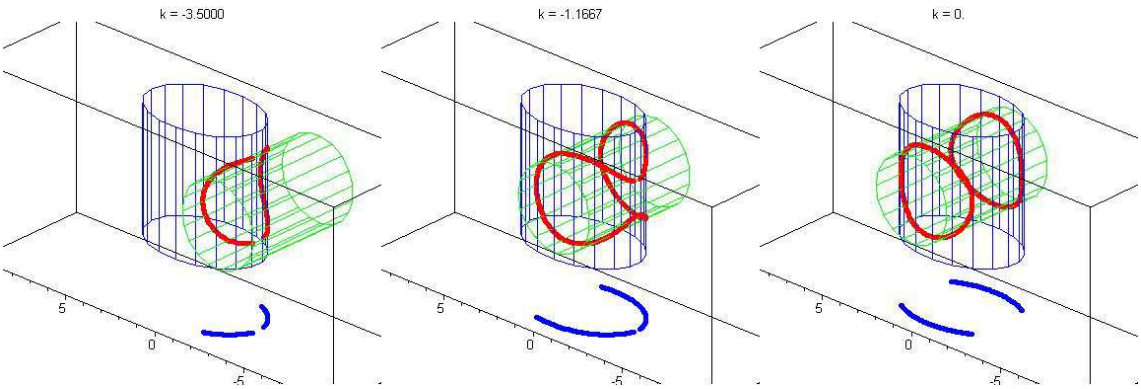


80. ábra.

8.3. Az elliptikus henger áthatása elliptikus hengerrel

```
>setoptions3d(axes=box,scaling=constrained,style=line);

>a:=3:b:=2:h1:=6:
>c:=2:d:=2:h2:=6:

>henger1:=plot3d([a*cos(u),b*sin(u),v],u=0..2*Pi,v=0..h1,
```

```
grid=[20,2],color=blue):

>henger2:=animate(plot3d,[[c*cos(u)+k,v-h2/2,d*sin(u)+d+1],
u=0..2*Pi,v=0..h2,grid=[20,2],color=green],k=-7..7):

>hengeregy:=x^2/a^2+y^2/b^2=1:

>hengerhely:=subs({x=c*cos(u)+k,y=v-h2/2},hengeregy):

>vm:=solve(hengerhely,v):

>ag1:=animate(spacecurve,[[c*cos(u)+k,vm[1]-h2/2,d*sin(u)+d+1],
u=0..2*Pi,color=red,thickness=3,numpoints=151],k=-7..7):

>ag2:=animate(spacecurve,[[c*cos(u)+k,vm[2]-h2/2,d*sin(u)+d+1],
u=0..2*Pi,color=red,thickness=3,numpoints=151],k=-7..7):

>agv1:=animate(spacecurve,[[c*cos(u)+k,vm[1]-h2/2,-3],
u=0..2*Pi,color=blue,thickness=3,numpoints=151],k=-7..7):

>agv2:=animate(spacecurve,[[c*cos(u)+k,vm[2]-h2/2,-3],
u=0..2*Pi,color=blue,thickness=3,numpoints=151],k=-7..7):

>display([henger1,henger2,ag1,ag2,agv1,agv2],view=[-10..10,-3..3,
-3..6]); (81. ábra)
```



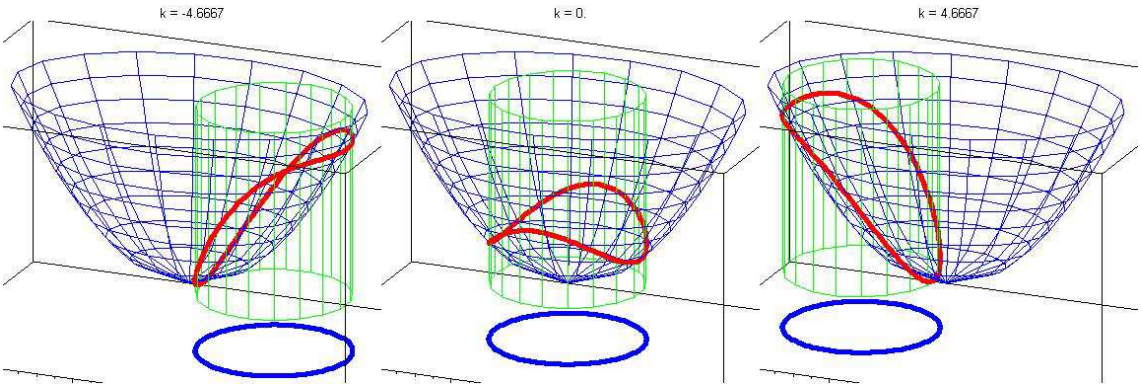
81. ábra.

8.4. Az elliptikus paraboloid áthatási görbéi

8.4.1. Az elliptikus paraboloid áthatása elliptikus hengerrel

```
>a:=3:b:=2:h1:=10:
>c:=4:d:=4:h2:=10:
```

```
>parab:=plot3d([a*sqrt(u)*cos(v),b*sqrt(u)*sin(v),u],v=0..2*Pi,
u=0..h1,grid=[20,10],color=blue):

>henger:=animate(plot3d,[[c*cos(u)+k,d*sin(u),v],u=0..2*Pi,
v=0..h2,grid=[20,2],color=green],k=-7..7):

>parabegy:=x^2/a^2+y^2/b^2=z:

>parabhely:=subs({x=c*cos(u)+k,y=d*sin(u),z=v},parabegy):

>vm:=solve(parabhely,v):

>ag:=animate(spacecurve,[[c*cos(u)+k,d*sin(u),vm],u=0..2*Pi,
color=red,thickness=3,numpoints=151],k=-7..7):

>agv:=animate(spacecurve,[[c*cos(u)+k,d*sin(u),-3],u=0..2*Pi,
color=blue,thickness=3,numpoints=151],k=-7..7):

>display([parab,henger,ag,agv],view=[-12..12,-8..8,-3..10]);
(82. ábra)
```



82. ábra.

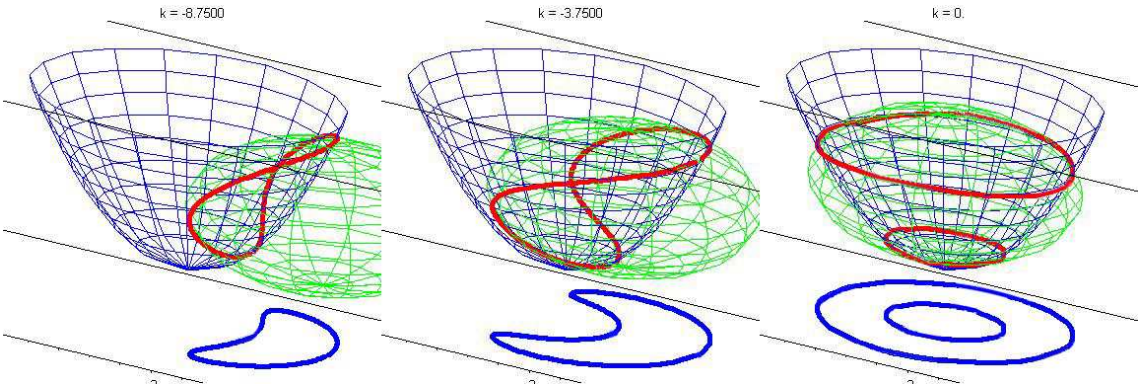
8.4.2. Az elliptikus paraboloid áthatása ellipszoiddal

```
>a:=3:b:=2:h1:=10:
>c:=8:d:=6:e:=4:

>parab:=plot3d([a*sqrt(u)*cos(v),b*sqrt(u)*sin(v),u],v=0..2*Pi,
u=0..h1,grid=[20,10],color=blue):

>ellipszoid:=animate(plot3d,[[c*cos(t)*sin(f)+k,d*sin(t)*sin(f),
e*cos(f)+e+1],t=0..2*Pi,f=0..Pi,grid=[15,15],color=green],
k=-15..15):
```

```
>parabegy:=x^2/a^2+y^2/b^2=z:

>parabhely:=subs({x=c*cos(t)*sin(f)+k,y=d*sin(t)*sin(f),
z=e*cos(f)+e+1},parabegy):

>tm:=solve(parabhely,t):

>ag1:=animate(spacecurve,[[c*cos(tm[1])*sin(f)+k,d*sin(tm[1])*
sin(f),e*cos(f)+e+1],f=0..Pi,color=red,thickness=3,numpoints=151],
k=-15..15):
```

Az előbbi `parabegy` egyenletbe behelyettesítve az ellipszoid paraméteres egyenletének tagjait, majd ezt t-re megoldva 4 gyököt kapunk. Az áthatási görbét 4 részből állítjuk elő, ezek az `ag1,ag2,ag3,ag4` görbék, amelyek csak annyiban térnek el, hogy paraméteres egyenletükben a megfelelő helyre a megfelelő gyököt (`tm[1],tm[2],...`) helyettesítjük.
Hasonlóan kapjuk meg az áthatási görbe vetületének képét is:

```
>ag1v:=animate(spacecurve,[[c*cos(tm[1])*sin(f)+k,d*sin(tm[1])*
sin(f),-3],f=0..Pi,color=blue,thickness=3,numpoints=151],k=-15..15):

>display([parab,ellipszoid,ag1,ag2,ag3,ag4,ag1v,ag2v,ag3v,ag4v]);
(83. ábra)
```

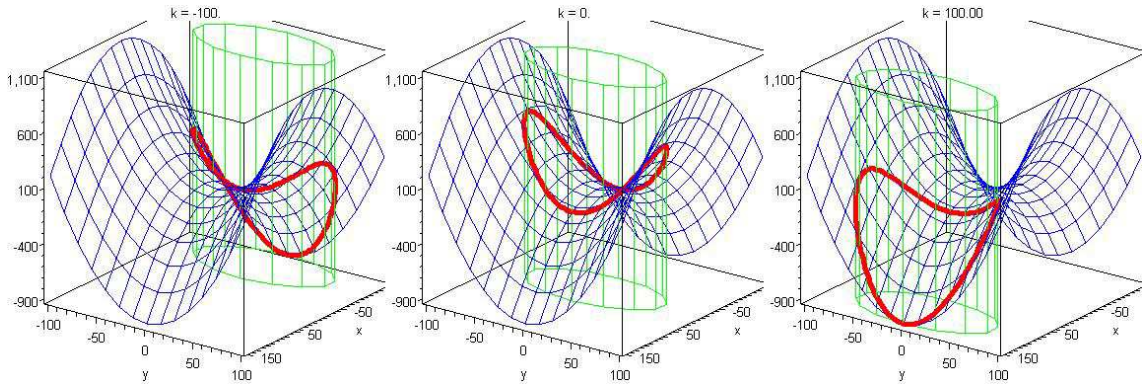


83. ábra.

8.5. A hiperbolikus paraboloid áthatása elliptikus hengerrel

```
>a:=5:b:=3:
>c:=50:d:=70:h:=2000:m:=900: #m az eltoláshoz

>hippar:=plot3d(y^2/b^2-x^2/a^2,x=-150..150,y=-100..100,color=blue,
grid=[15,15]):

>henger:=animate(plot3d,[[c*cos(u)+k,d*sin(u),v-m],u=0..2*Pi,v=0..h,
```

```
color=green,grid=[15,2]],k=-100..100):

>hipegy:=y^2/b^2-x^2/a^2=z:

>hipegyhely:=subs({x=c*cos(u)+k,y=d*sin(u),z=v-m},hipegy):

>vm:=solve(hipegyhely,v):

>ag:=animate(spacecurve,[[c*cos(u)+k,d*sin(u),vm-m],u=0..2*Pi,
color=red,thickness=3,numpoints=151],k=-100..100):

>display([hippar,henger,ag]); (84. ábra)
```



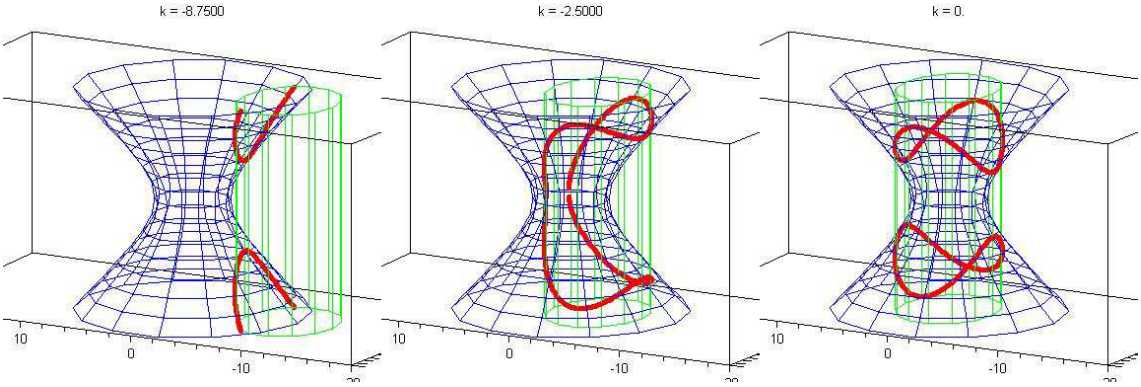
84. ábra.

8.6. Az egyköpenyű hiperboloid áthatási görbéi

8.6.1. Az egyköpenyű hiperboloid áthatási görbéje elliptikus hengerrel

```
>a:=3:b:=2:h1:=3:
>c:=4:d:=5:h2:=18:m:=9:

>hipegykop:=plot3d([a*sqrt(1+u^2)*cos(v),a*sqrt(1+u^2)*sin(v),
h1*u],v=0..2*Pi,u=-h1..h1,color=blue,grid=[15,15]):

>henger:=animate(plot3d,[[c*cos(u)+k,d*sin(u),v-m],u=0..2*Pi,v=0..
h2,color=green,grid=[15,2]],k=-15..15):

>hipegy:=x^2/a^2+y^2/b^2-z^2/h1^2=1:

>hiphely:=subs({x=c*cos(u)+k,y=d*sin(u),z=v-m},hipegy):

>vm:=solve(hiphely,v):
```

```
>ag1:=animate(spacecurve,[[c*cos(u)+k,d*sin(u),vm[1]-m],u=0..2*Pi,
color=red,thickness=3,numpoints=151],k=-15..15):

>ag2:=animate(spacecurve,[[c*cos(u)+k,d*sin(u),vm[2]-m],u=0..2*Pi,
color=red,thickness=3,numpoints=151],k=-15..15):

>display([hipegykop,henger,ag1,ag2],view=[-20..20,-10..10,-9..9]);
(85. ábra)
```



85. ábra.

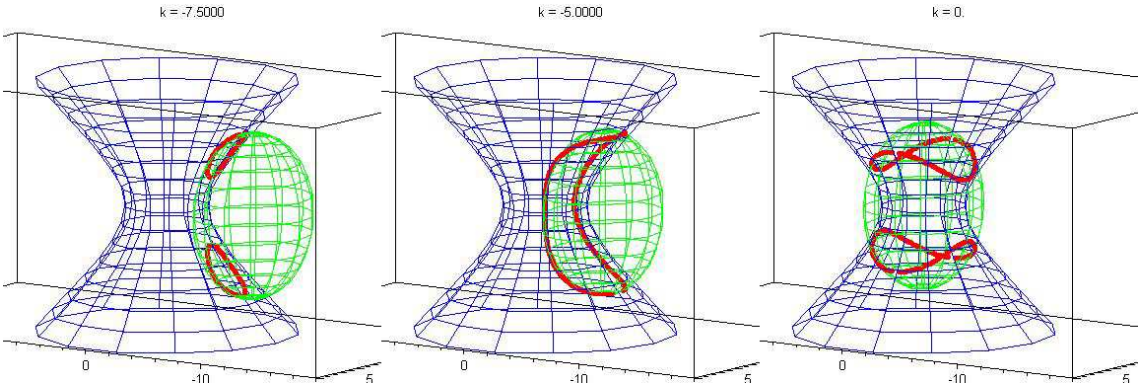
8.6.2. Az egyköpenyű hiperboloid áthatási görbéje ellipszoiddal

```
>a:=3:b:=2:h1:=3:
>c:=4:d:=5:e:=6:

>hipegykop:=plot3d([a*sqrt(1+u^2)*cos(v),a*sqrt(1+u^2)*sin(v),
h1*u],v=0..2*Pi,u=-h1..h1,color=blue,grid=[15,15]):

>henger:=animate(plot3d,[[c*cos(t)*sin(f)+k,d*sin(t)*sin(f),e*
cos(f)],t=0..2*Pi,f=0..Pi,color=green,grid=[15,15]],k=-15..15):

>hipegy:=x^2/a^2+y^2/b^2-z^2/h1^2=1:

>hiphely:=subs({x=c*cos(t)*sin(f)+k,y=d*sin(t)*sin(f),
z=e*cos(f)},hipegy):

>tm:=solve(hiphely,t):

>ag1:=animate(spacecurve,[[c*cos(tm[1])*sin(f)+k,d*sin(tm[1])*
sin(f),e*cos(f)],f=0..Pi,color=red,thickness=3,numpoints=151],
k=-15..15):
```

Az áthatási görbét 4 részből rakjuk össze, a másik három rész képét is az előzőhöz hasonlóan kapjuk, csak a megfelelő `tm[i]` értéket írjuk a parancsba.

```
>display([hipegykop,henger,ag1,ag2,ag3,ag4],view=[-20..20,
```

```
-10..10,-9..9]); (86. ábra)
```

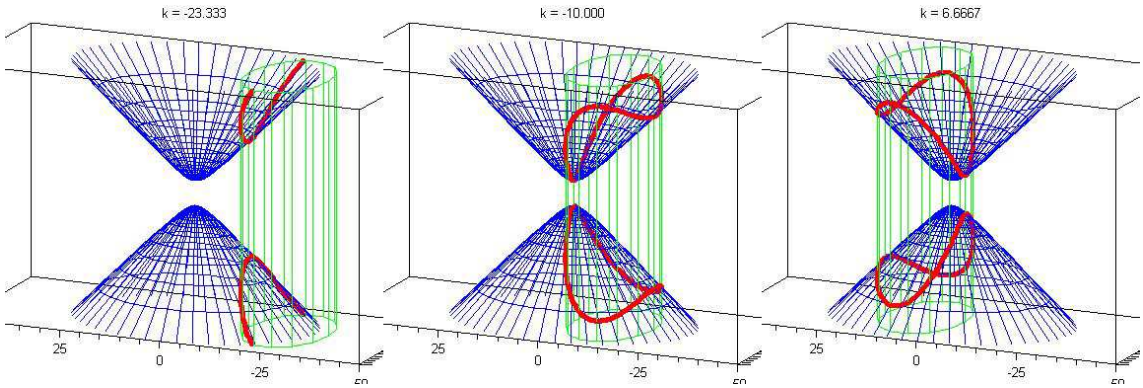


86. ábra.

8.7. A kétköpenyű hiperboloid áthatási görbéi

8.7.1. A kétköpenyű hiperboloid áthatási görbéje elliptikus hengerrel

```
>a:=3:b:=2:h1:=3:
>c:=10:d:=15:h2:=60:m:=30:

>hipketkop1:=plot3d([a*sinh(u)*cos(v),b*sinh(u)*sin(v),h1*cosh(u)],
v=0..Pi,u=-h1..h1,color=blue):
>hipketkop2:=plot3d([a*sinh(u)*cos(v),b*sinh(u)*sin(v),-h1*cosh(u)],
v=0..Pi,u=-h1..h1,color=blue):

>henger:=animate(plot3d,[[c*cos(u)+k,d*sin(u),v-m],u=0..2*Pi,v=0..
h2,color=green,grid=[15,2]],k=-40..40):

>hipegy:=x^2/a^2+y^2/b^2-z^2/h1^2=-1:

>hiphely:=subs({x=c*cos(u)+k,y=d*sin(u),z=v-m},hipegy):

>vm:=solve(hiphely,v):

>ag1:=animate(spacecurve,[[c*cos(u)+k,d*sin(u),vm[1]-m],u=0..2*Pi,
color=red,thickness=3,numpoints=151],k=-40..40):

>ag2:=animate(spacecurve,[[c*cos(u)+k,d*sin(u),vm[2]-m],u=0..2*Pi,
color=red,thickness=3,numpoints=151],k=-40..40):

>display([hipketkop1,hipketkop2,henger,ag1,ag2],view=[-50..50,
-20..20,-30..30]); (87. ábra)
```



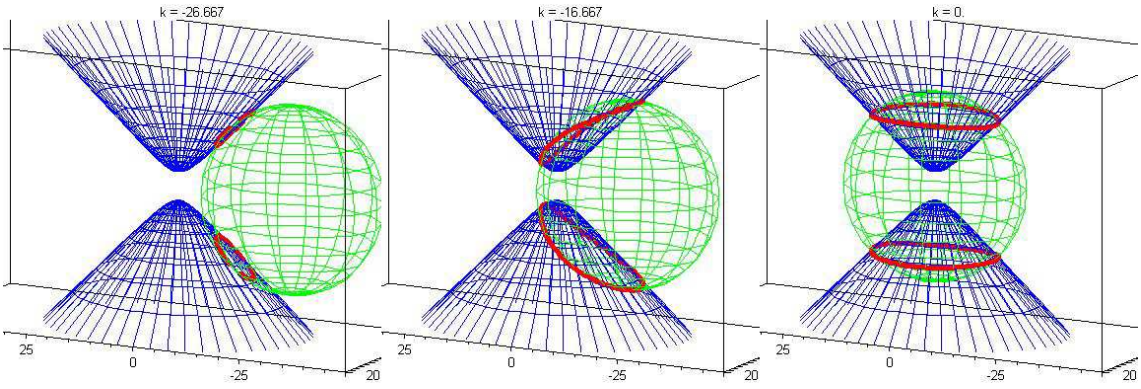
87. ábra.

8.7.2. A kétköpenyű hiperboloid áthatási görbéje ellipszoiddal

```
>a:=3:b:=2:h1:=3:
>c:=20:d:=15:e:=20:

>hipketkop1:=plot3d([a*sinh(u)*cos(v),b*sinh(u)*sin(v),h1*cosh(u)],
v=0..Pi,u=-h1..h1,color=blue):

>hipketkop2:=plot3d([a*sinh(u)*cos(v),b*sinh(u)*sin(v),-h1*cosh(u)],
v=0..Pi,u=-h1..h1,color=blue):

>ellipszoid:=animate(plot3d,[[c*cos(t)*sin(f)+k,d*sin(t)*sin(f),e*
cos(f)],t=0..2*Pi,f=0..Pi,color=green,grid=[15,15]],k=-40..40):

>hipegy:=x^2/a^2+y^2/b^2-z^2/h1^2=-1:

>hiphely:=subs({x=c*cos(t)*sin(f)+k,y=d*sin(t)*sin(f),z=e*cos(f)},
hipegy):

>fm:=solve(hiphely,f):

>ag1:=animate(spacecurve,[[c*cos(t)*sin(fm[1])+k,d*sin(t)*
sin(fm[1]),e*cos(fm[1])],t=0..2*Pi,color=red,thickness=3,numpoints=
151],k=-40..40):
```

Az áthatási görbét a korábbiakhoz hasonlóan most is 4 részből állítjuk elő (`ag1,2,3,4`).

```
>display([hipketkop1,hipketkop2,ellipszoid,ag1,ag2,ag3,ag4],
view=[-50..50,-20..20,-30..30]); (88. ábra)
```



88. ábra.

Irodalomjegyzék

[1] Maple User Manual, `http://www.maplesoft.com/documentation_center/`

[2] Wolfram MathWorld, `http://mathworld.wolfram.com/`

[3] André Heck: *Bevezetés a Maple használatába*, Szeged, 1999, JGYF Kiadó.

[4] Molnárka-Gergó-Wettl-Horváth-Kallós: *A Maple V és alkalmazásai*, Budapest, 1996, Springer Hungarica Kiadó.